MASTER EN SCIENCES MATHÉMATIQUES

Une étude de techniques de comparaison moléculaire

Culot, Patrick; Gillo, Xavier

Award date:
1987

Facultés Universitaires N. D. de la Paix

Namur

Faculté des sciences

UNE ETUDE DE TECHNIQUES DE COMPARAISON MOLECULAIRE

Mémoire présenté pour l'obtention du grade
de Licencié en Sciences mathématiques
par

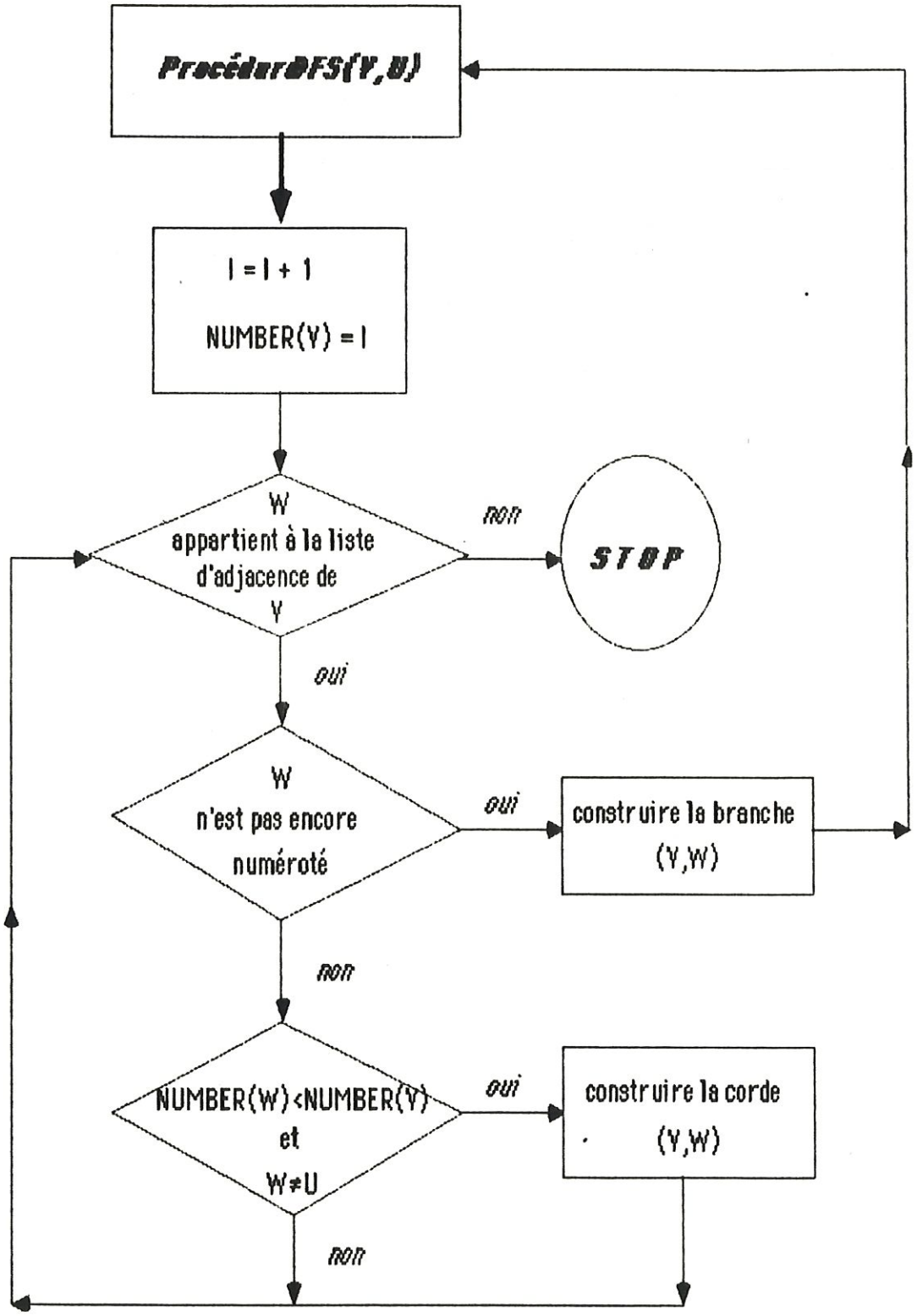
Promoteur : **V.H. NGUYEN**
Directeur : **D.P. VERCAUTEREN**

CULOT PATRICK
GILLO XAVIER

Année académique : 1986-1987

Erratum de la page V.18

I:=0
DFS(S,0)



Facultés Universitaires Notre-Dame de la Paix
Faculté des Sciences
rue de Bruxelles 61, B-5000 **NAMUR**
Tél. 081-22.90.61 Télex 59222 facnam-b Téléfax 081-23.03.91

UNE ETUDE DE TECHNIQUES DE COMPARAISON MOLECULAIRE

CULOT Patrick
GILLO Xavier

Résumé

Les méthodes de comparaison moléculaire, souvent utilisées en pharmacologie, essaient de relier la structure des composés à leur activité afin de prédire certaines propriétés et effets.

L'introduction de l'énergie conformationnelle dans celles-ci est particulièrement importante et à notre connaissance aucune "recette universelle" n'a été proposée à ce jour.

Dans ce cadre, nous avons développé trois techniques de comparaison moléculaire tenant compte de notions d'énergie (mécanique moléculaire) et une quatrième considérant plusieurs liaisons interatomiques simples comme axes de rotation.

En outre, nous avons élaboré plusieurs algorithmes vérifiant la connexité de la molécule ainsi que l'admissibilité des liaisons prises comme axe de rotation.

Abstract

Molecular fitting techniques largely used in molecular pharmacology, try to relate the structure of the coumpounds to their activity in order to predict propeties and effects of related drugs

Considerations of conformationnel energy notions in these methods are particulary important but to our knowledge no "universal recipe" has been proposed so far.

Therefore, we develoled three different fitting techniques including energy notions (molecular mechanics) and a fourth one allowing rotation around multiple interatomic simple bounds.

Moreover, we implemented an algorithm to verify the connectivity of the molecule and the acceptability for a bond to rotate.

Mémoire de licence en Sciences mathématiques
Juin 1987
Promoteur : V.H. NGUYEN
Directeur : D.P. VERCAUTEREN

Ce mémoire a pu être réalisé grâce à une étroite collaboration entre les départements de Chimie et de Mathématique des Facultés Universitaires Notre-Dame de la Paix.

Nous tenons à remercier monsieur le Professeur F. Durant qui nous a accueillis chaleureusement au Laboratoire de Chimie Moléculaire Structurale et Radiocristallographie (C.M.S.) et monsieur le Professeur V.H. Nguyen, de l'Unité d'Optimisation, qui nous a aidés tout au long de l'élaboration de notre mémoire.

Nous exprimons notre reconnaissance au Laboratoire C.M.S. et particulièrement au Docteur D.P. Vercauteren pour leur constante disponibilité et l'intérêt qu'ils ont manisfesté pour ce travail de fin d'études.

Nos remerciements sont également adressés à l'Unité d'Optimisation du département de Mathématique et, plus particulièrement, au Docteur J.J. Strodiot qui nous ont prodigués beaucoup de conseils.

Avant-propos

Les méthodes de comparaison moléculaire, souvent utilisées en pharmacologie, essaient de relier la structure des composés à leur activité afin de prédire certaines propriétés et effets.

Nous parlerons dans un premier temps, de deux techniques de comparaison moléculaire utlisées actuellement au département de Chimie.

La première envisage une superposition de deux molécules, l'une étant considérée comme molécule de référence (choisie parce que possédant une activité pharmacologique), l'autre étant déplacée entièrement sur la molécule de référence.

La deuxième méthode tient compte de la flexibilité de la seconde molécule et permet d'effectuer une superposition en faisant pivoter un ou plusieurs groupements autour d'une ou plusieurs liaisons prises successivement comme axes de rotation.

La deuxième méthode, appelée fitting flexible, ne tient pas compte de l'énergie interne de la molécule. Ce concept est cependant important car à un changement de conformère (une molécule dans une autre conformation) est associé une variation d'énergie qui ne peut être réalisable de part la proximité trop forte de deux groupements volumineux par exemple.

De plus, le déplacement d'une partie de la molécule ne s'effectue qu'autour d'une liaison simple prise comme axe de rotation.

Dans ce travail, nous soulevons le problème de l'énergie qui est envisagé sous différentes méthodes et traitons d'une technique utilisant plusieurs liaisons simples comme axes de rotation.

Nous développons également des algorithmes vérifiant la connexité de la molécule ainsi que l'admissibilité des liaisons prises comme axes de rotation.

Table des matières

Chapitre III: **Fitting flexible avec énergie sur un axe de rotation**

Chapitre IV: **Fitting flexible sur plusieurs axes de rotation**

Chapitre V: **Vérifications**

Chapitre 1

Caractérisation et évaluation prédictive par des techniques de comparaison moléculaire ou fitting.

Dans ce chapitre, nous expliquons brièvement le rôle des techniques de comparaison moléculaire en pharmacologie [1].

Beaucoup de propriétés physico-chimiques d'un composé conduisent à des applications utiles et sont donc largement étudiées afin d'obtenir de plus en plus de produits spécifiques. Toutefois, ces propriétés sont rarement liées à la structure microscopique du composé au moyen de fonctions liant les propriétés macroscopiques aux caractéristiques moléculaires quantitatives. Des propriétés telles que la stabilité thermodynamique, la réactivité chimique, l'activité biologique ou pharmacologique d'un composé, les caractéristiques mécaniques et les phases physiques des matériaux macro-moléculaires sont liées de manière complexe à des caractéristiques moléculaires intrinsèques telles que l'énergie totale, les forces de liaison entre atomes, la distribution intermoléculaire des charges et le champ électrique de la molécule, la possibilité de rotations internes ...

En raison de la complexité de la nature des liaisons, il n'est pas aisé en pharmacologie, lorsque l'on recherche de nouveaux produits ou des produits plus spécifiques, de réaliser des évaluations prédictives à propos de l'effet (ou des propriétés) de ceux-ci. Il est donc fort utile d'avoir en main des moyens d'effectuer des études comparatives des caractéristiques moléculaires intrinsèques et de relever des similitudes à la fois structurelles et "conformationnelles" entre des composés dits actifs.

L'usage combiné de méthodes d'analyse "conformationnelle" (en général de résolution de structure) et de procédures d'évaluation d'ajustement stérique peut mener, dans une série de molécules ayant une activité spécifique, à l'extraction de caractéristiques moléculaires communes aux meilleurs composés et exprimables sous forme de paramètres quantitatifs (appelés descripteurs) ou d'effets pratiques. Ces ensembles de paramètres (ou d'effets) peuvent être corrélés avec des valeurs d'activité et peuvent être calculés (ou vérifiés) pour un nouveau composé afin de prévoir ses propriétés. Ils pourraient ainsi mener à la définition d'un nouveau composé.

Dans la recherche de descripteurs moléculaires caractérisant une molécule biologiquement active, des quantités physiques telles que la forme de la molécule dans les conformations les plus probables, sont les paramètres les plus fréquemment utilisés à la fois dans les problèmes d'identification moléculaire et les études de rapports entre la structure et l'activité.

Les premiers problèmes sont liés à l'identification des propriétés physico-chimiques essentielles pour l'interaction intermoléculaire entre les médicaments et leurs sites réceptifs. Les descripteurs peuvent fournir des modèles appropriés pour ces interactions et pertubations "conformationnelles" qui accompagnent le remplissage du site réceptif.

Dans le second type de problèmes, les descripteurs apparaissent appropriés pour exprimer les ressemblances entre des composés actifs et pour rechercher des corrélations menant à prédire l'activité d'un nouveau composé. Leur utilité, dans ce cas, est basée sur l'hypothèse que dans une série de médicaments ayant une même activité spécifique mais un mode d'action inconnu, la plupart des molécules actives peuvent aisément adopter des conformations semblables qui s'adaptent bien aux sites réceptifs. Cependant, il est généralement nécessaire de connaître non seulement quelles sont les régions spatiales occupées par les atomes des composés mais aussi par "quoi" ces régions sont occupées. Une réponse à cette dernière question est donnée par une description de la distribution électronique dans la molécule calculée, par exemple, par des méthodes de mécanique quantique.

Après une première évaluation des propriétés structurales et électroniques des composés considérés isolément, des techniques de superposition stérique et électronique deviennent essentielles pour essayer de rechercher des caractéristiques communes expliquant leur activité. Comme nous l'avons laissé sous-entendre précédemment, une première approche serait basée sur la comparaison de volume moléculaire global occupé; une seconde, plus précise surtout si l'on utilise des données électroniques, sur la correspondance atomique.

Les procédures développées par la suite permettent d'évaluer la ressemblance moléculaire lorsque l'on connaît les parties qui sont comparées ou lorsque l'on peut faire quelque hypothèse sur la correspondance atomique.

Chapitre II

Fitting

Nous exposons deux techniques de comparaison moléculaire, l'une déplaçant l'entièreté de la molécule et l'autre, une partie de celle-ci.

Introduction

Les techniques de comparaison moléculaire permettent de dégager les analogies ou les différences entre deux molécules, une molécule de référence et une molécule mobile.

Dans cet ouvrage, nous traiterons de techniques effectuant la superposition par une minimisation de la somme des distances euclidiennes au carré entre les atomes à comparer des deux molécules.

Cette minimisation permet un déplacement de la molécule mobile qui se superpose sur la molécule de référence. Ce déplacement de la molécule mobile concerne soit l'entièreté de celle-ci, c'est le *fitting rigide*, soit une partie de celle-ci, c'est le *fitting flexible*.

2.1 Notations

Une molécule étant un ensemble d'atomes, nous poserons:

$X=\{x_i,\ i=1,...,R\}$	la molécule de référence constituée de **R** atomes x_i;
$Y=\{y_j,\ j=1,...,B\}$	la molécule mobile constituée de **B** atomes y_j;
$(x_{i1},x_{i2},x_{i3})^t$	les coordonnées de l'atome x_i de la molécule de référence;
$(y_{j1},y_{j2},y_{j3})^t$	les coordonnées de l'atome y_j de la molécule mobile.

Nous considérons, sans perte de généralité, que les atomes à comparer des molécules de référence et mobile sont les **E** premiers atomes de chacune des deux molécules. Nous écrirons:

$AFE=\{x_i \in X,\ i=1,...,E\}$	l'ensemble des atomes à superposer de la molécule de référence;
$AME=\{y_i \in Y,\ i=1,...,E\}$	l'ensemble des atomes à superposer de la molécule mobile.

Nous considérons que ces 2 ensembles ne sont pas vides.
Nous superposons le i° élément de AME avec le i° élément de AFE, i=1,...,E.

Pour les rotations de la molécule mobile, nous supposons que l'axe passe par l'origine du système de référence de cette molécule et noterons:

$\vec{e}$	le vecteur "axe de rotation";
e_1,e_2,e_3	les cosinus directeurs du vecteur $\vec{e}$;
δ	l'angle de rotation.

2.2 Fitting rigide

Le fitting rigide, développé par Nyburg en 1974 [2] dans un programme appelé BMFIT (Best Molecular FIT) [3], fut la première technique de comparaison moléculaire.

Il effectue la superposition en déplaçant toute la molécule mobile pour la faire coïncider avec la molécule de référence.

a) Hypothèse

Nous supposerons que les ensembles AFE et AME contiennent au moins trois éléments.

En effet, pour définir la situation d'un objet dans l'espace, il est nécessaire de fixer la position de trois points non alignés de cet objet.

b) Problème

La situation de la molécule dans l'espace est déterminée par trois atomes non alignés, soit neuf paramètres non indépendants puisque les coordonnées de ces atomes sont liées par trois relations exprimant l'invariabilité de la distance entre ces atomes. Définir la situation d'une molécule nécessite donc la connaissance de six paramètres: trois paramètres indépendants définissant la position d'un atome de la molécule et trois paramètres indépendants déterminant l'orientation de la molécule autour de l'atome précédent.

Le déplacement de la molécule mobile consiste en une translation et une rotation.

La translation fait coïncider le premier élément de AME avec le premier élément de AFE.

La rotation permet de faire concorder les autres éléments de AME avec les autres éléments de AFE.

Nous effectuerons successivement une translation de la molécule de référence pour déplacer le premier élément de AFE en l'origine et une translation de la molécule mobile pour déplacer le premier élément de AME en l'origine.

Cette superposition effectuée, le problème du fitting rigide consiste alors en la recherche de l'axe et de l'angle de rotation permettant de faire coïncider au mieux les atomes à comparer. Nous recherchons donc l'axe et l'angle de rotation optimaux minimisant la somme des distances au carré entre les éléments de AME et AFE.

Le problème s'écrit:

$$\text{minimiser } F_1(\delta, e_1, e_2, e_3) = \Sigma_{i=1,E} \{ [x_{i1} - (Ay_i)_1]^2 + [x_{i2} - (Ay_i)_2]^2 + [x_{i3} - (Ay_i)_3]^2 \} \qquad (2.1)$$

où A est la matrice exprimant la rotation d'un angle δ de toute la molécule autour de l'axe de rotation de vecteur $\vec{e}$.

L'axe et l'angle de rotation étant inconnus, le problème est un problème de minimisation d'une fonction non linéaire à 4 variables: δ, l'angle de rotation et e_1, e_2, e_3 les cosinus directeurs du vecteur $\vec{e}$.

La matrice orthogonale de rotation A est donnée par:

$$A = (\cos\delta)\, I + (1 - \cos\delta)\, D + (\sin\delta)\, C \qquad (2.2)$$

$$\text{où } D = \begin{pmatrix} e_1^2 & e_1e_2 & e_1e_3 \\ e_2e_1 & e_2^2 & e_2e_3 \\ e_3e_1 & e_3e_2 & e_3^2 \end{pmatrix}$$

$$C = \begin{pmatrix} 0 & -e_3 & 0 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{pmatrix}$$

I = matrice identité 3 x 3

Nous allons simplifier l'expression de la matrice de rotation.

Si on suppose l'angle δ petit, on a:

$\cos \delta \approx 1$

$\sin \delta \approx \delta$

D'où découle une nouvelle matrice:

$$\mathbf{A' = I + \delta\, C} \qquad (2.3)$$

De plus, par l'expression des cosinus directeurs: $e_1^2 + e_2^2 + e_3^2 = 1$, nous réduisons le nombre de variables.

En effet: $\delta = [\,(\delta e_1)^2 + (\delta e_2)^2 + (\delta e_3)^2\,]$

$$e_1 = \frac{\delta e_1}{\delta},\quad e_2 = \frac{\delta e_2}{\delta},\quad e_3 = \frac{\delta e_3}{\delta}$$

On obtient le problème:

$$\textbf{minimiser } F_2(\delta e_1, \delta e_2, \delta e_3) = \Sigma_{i=1,E}\,[\,[x_{i1} - (A'y_i)_1]^2 + [x_{i2} - (A'y_i)_2]^2 + [x_{i3} - (A'y_i)_3]^2\,] \qquad (2.4)$$

où la matrice A' est donnée par l'expression (2.3), la fonction objectif F_2 dépend de trois variables: $\delta e_1, \delta e_2, \delta e_3$.

Nous obtenons la solution du problème (2.4) en résolvant le système de 3 équations à 3 inconnues δe_1, δe_2, δe_3:

$$[(\mathrm{Tr}\, M)\, I - M] \begin{pmatrix} \delta e_1 \\ \delta e_2 \\ \delta e_3 \end{pmatrix} = \begin{pmatrix} \Sigma_{i=1,E}\,(x_{i3} y_{i2} - x_{i2} y_{i3}) \\ \Sigma_{i=1,E}\,(x_{i1} y_{i3} - x_{i3} y_{i1}) \\ \Sigma_{i=1,E}\,(x_{i2} y_{i1} - x_{i1} y_{i2}) \end{pmatrix} \qquad (2.5)$$

$$\text{où } M = \begin{pmatrix} \Sigma_{i=1,E}\, y_{i1}^2 & \Sigma_{i=1,E}\, y_{i1} y_{i2} & \Sigma_{i=1,E}\, y_{i1} y_{i3} \\ \Sigma_{i=1,E}\, y_{i2}\, y_{i1} & \Sigma_{i=1,E}\, y_{i2}^2 & \Sigma_{i=1,E}\, y_{i2} y_{i3} \\ \Sigma_{i=1,E}\, y_{i3} y_{i1} & \Sigma_{i=1,E}\, y_{i3} y_{i2} & \Sigma_{i=1,E}\, y_{i3}^2 \end{pmatrix}$$

$$\text{Tr } M = \Sigma_{i=1,E}\, y_{i1}^2 + y_{i2}^2 + y_{i3}^2$$

I = matrice identité 3 x 3

Soient $\delta^* e_1^*$, $\delta^* e_2^*$, $\delta^* e_3^*$ les solutions du système d'équations (2.5).

Par la relation des cosinus directeurs, $e_1^2 + e_2^2 + e_3^2 = 1$, nous obtenons les solutions optimales du problème (2.4):

$$\delta^* = [\,(\delta^* e_1^*)^2 + (\delta^* e_2^*)^2 + (\delta^* e_3^*)^2\,] \qquad (2.6)$$

$$e_1^* = \frac{\delta^* e_1^*}{\delta^*},\quad e_1^* = \frac{\delta^* e_2^*}{\delta^*},\quad e_3^* = \frac{\delta^* e_3^*}{\delta^*}$$

La solution du problème (2.1) est alors obtenue en minimisant successivement la fonction objectif F_2 (problème (2.4)) jusqu'à ce que la fonction F_1 ne diminue plus que d'une certaine tolérance.

c) Inconvénient de la méthode

Cette méthode déplace globalement la molécule mobile. Elle considère donc celle-ci comme rigide et ne tient nullement compte d'une flexibilité éventuelle de cette molécule permettant de bouger certaines parties de celle-ci.

d) Théorème d'Euler [4]

Ce théorème montre qu'il est possible de trouver un axe et un angle de rotation minimisant la somme des distances au carré entre les atomes à comparer.

Considérons un corps rigide ayant un point fixe O autour duquel peut pivoter ce corps. Prenons deux positions possibles du corps, nous les appellerons position P et position Q.

Nous allons voir qu'il est possible de faire passer le corps rigide de la position P à la position Q par une rotation autour d'un axe passant par le point fixe O.

Le *théorème d'Euler* s'énonce:

La rotation autour d'un point est toujours équivalente à une rotation autour d'une droite passant par ce point.

Preuve

Soient OA, OB deux droites, dans la position P, passant par le point fixe. Elles sont fixes dans le corps rigide et bougent avec lui.

Soient OA', OB' ces droites dans la position Q.

Considérons le plan π perpendiculaire au plan (AOA') et bissectrice de l'angle $\widehat{AOA'}$,

le plan π' perpendiculaire au plan (BOB') et bissectrice de l'angle $\widehat{BOB'}$.

Si les plans π et π' ne coïncident pas, nous noterons OC la droite d'intersection de ces 2 plans, sinon OC est la droite d'intersection des plans (OAB) et (OA'B').

Par la position de la droite OC par rapport aux droites OA, OB, on a: les angles $\widehat{AOC}$ et $\widehat{BOC}$ sont respectivement égaux aux angles $\widehat{A'OC}$ et $\widehat{B'OC}$.

D'où: le système (OABC) tourne autour de O de telle façon que les droites OA, OB se déplacent, respectivement, aux positions OA', OB' et que la droite OC ne change pas de position.

Donc, la droite OC n'est pas affectée par le déplacement du corps.

C'est-à-dire le déplacement est une rotation autour de la droite OC.

e) Matrice de rotation

Théorème [5]

Considérons une rotation d'angle Θ autour du vecteur unitaire $\vec{e} = (e_1, e_2, e_3)$ passant par l'origine d'un système de référence.

La matrice orthogonale de rotation exprimant la transformation des coordonnées est:

$$A = (\cos \delta) I + (1 - \cos \delta) D + (\sin \delta) C$$

$$\text{où } D = \begin{pmatrix} e_1^2 & e_1 e_2 & e_1 e_3 \\ e_2 e_1 & e_2^2 & e_2 e_3 \\ e_3 e_1 & e_3 e_2 & e_3^2 \end{pmatrix}$$

$$C = \begin{pmatrix} 0 & -e_3 & 0 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{pmatrix}$$

$$I = \text{matrice identité } 3 \times 3$$

Preuve

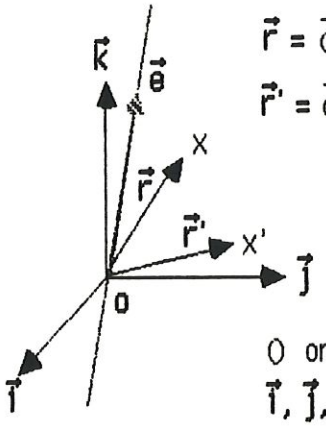
1) Montrons $A = (\cos \delta) I + (1 - \cos \delta) D + (\sin \delta) C$.

Pour déterminer la transformation des coordonnées, on considère la rotation d'angle Θ d'un vecteur $\vec{r}$ autour de l'axe. On pourra alors appliquer la rotation aux trois axes de coordonnées pour obtenir un nouveau système de référence et ainsi, en déduire la transformation des coordonnées.

Soient $\vec{r}$ un vecteur,

$\vec{r}'$ le vecteur $\vec{r}$ obtenu après une rotation d'angle Θ autour de l'axe de vecteur unitaire $\vec{e}$.

Notons $\vec{r} = \overrightarrow{OX}$



$\vec{r} = \overrightarrow{OX}$ un vecteur

$\vec{r}' = \overrightarrow{OX}'$ le vecteur $\vec{r}$ obtenu après une rotation d'angle Θ autour de l'axe de vecteur unitaire $\vec{e}$

O origine des coordonnées

$\vec{i}, \vec{j}, \vec{k}$ axes de coordonnées

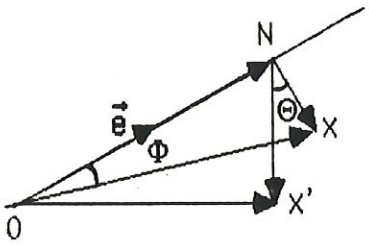
Décomposer $\overrightarrow{OX}$ en une somme de deux vecteurs dont un le long de l'axe: $\overrightarrow{OX} = \overrightarrow{ON} + \overrightarrow{NX}$

où N est un point appartenant à l'axe de rotation.

On a $\overrightarrow{ON}$ reste inchangé par la rotation;

$\overrightarrow{NX}'$ est le vecteur $\overrightarrow{NX}$ obtenu après la rotation d'angle Θ.

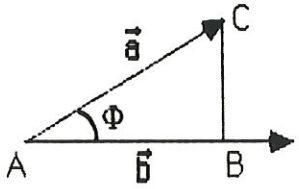
C'est-à-dire $\vec{r}' = \overrightarrow{OX}' = \overrightarrow{ON} + \overrightarrow{NX}'$



$\overrightarrow{OX}$ un vecteur

$\overrightarrow{OX}'$ le vecteur $\overrightarrow{OX}$ obtenu après une rotation d'angle Θ autour de l'axe de vecteur unitaire $\vec{e}$

Rappel: la composante du vecteur $\vec{a}$ le long de la direction de vecteur $\vec{b}$.



$AB = AC \cos\Phi$ (relation des triangles rectangles)

$= \|\vec{a}\| \cos\Phi$

$= \dfrac{\|\vec{a}\| \, \vec{a}\,\vec{b}}{\|\vec{a}\| \, \|\vec{b}\|}$ (définition du produit scalaire)

$= \dfrac{\vec{a}\,\vec{b}}{\|\vec{a}\|}$

Dès lors: $\overrightarrow{ON} = \dfrac{(\vec{r}\,\vec{e})\,\vec{e}}{\|\vec{e}\|}$

$= (\vec{r}\,\vec{e})\,\vec{e}$ ($\vec{e}$ est un vecteur unitaire)

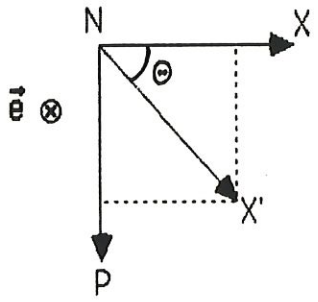
et $\overrightarrow{NX} = \vec{r} - (\vec{r}\,\vec{e})\,\vec{e}$ ($\vec{r} = \overrightarrow{ON} + \overrightarrow{NX}$)

Soit $\overrightarrow{NP} = \vec{e} \times \vec{r}$

Par la définition du produit cartésien:

$\overrightarrow{NP}$ est orthogonal à $\vec{e}$ et à $\vec{r}$

$\|\overrightarrow{NP}\| = \|\vec{e}\|\,\|\vec{r}\|\sin\Theta$



$\overrightarrow{NX'}$ est le vecteur $\overrightarrow{NX}$ obtenu après une rotation d'angle Θ autour de l'axe de vecteur unitaire $\vec{e}$

$\overrightarrow{NP} = \vec{e} \times \vec{r}$

On a $\|\overrightarrow{NX}\| = \|\vec{r}\|\sin\Theta$ (relation des triangles rectangles)

$= \|\vec{r}\|\,\|\vec{e}\|\sin\Theta$

$= \|\overrightarrow{NP}\|$ ($\vec{e}$ est un vecteur unitaire)

De plus, $\overrightarrow{NX'}$ est le vecteur $\overrightarrow{NX}$ obtenu après rotation d'un angle Θ

D'où $\|\overrightarrow{NX'}\| = \|\overrightarrow{NX}\|$

Dès lors, $\|\overrightarrow{NX'}\| = \|\overrightarrow{NX}\| = \|\overrightarrow{NP}\|$

Exprimons $\overrightarrow{NX'}$ en fonction de $\overrightarrow{NX}$ et $\overrightarrow{NP}$

$\overrightarrow{NX'} = \cos\Theta\ \overrightarrow{NX} + \sin\Theta\ \overrightarrow{NP}$

$= \cos\Theta\,[\,\vec{r} - (\vec{r}.\vec{e}).\vec{e}\,] + \sin\Theta\,(\vec{e} \times \vec{r})$

Nous obtenons l'expression de $\vec{r}'$ en fonction de $\vec{r}$:

$\vec{r}' = \overrightarrow{ON} + \overrightarrow{NX'}$

$= (\vec{r}.\vec{e}).\vec{e} + \cos\Theta\,[\,\vec{r} - (\vec{r}.\vec{e}).\vec{e}\,] + \sin\Theta\,(\vec{e} \times \vec{r})$

Rappel:

$$\vec{a} \cdot \vec{b} = \vec{a}^t \vec{b} = (a_1\ a_2\ a_3) \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$(\vec{a} \cdot \vec{b}) \cdot \vec{c} = (\vec{a}^t \vec{b})\,\vec{c} = \begin{pmatrix} (a_1b_1 + a_2b_2 + a_3b_3)\, e_1 \\ (a_1b_1 + a_2b_2 + a_3b_3)\, e_2 \\ (a_1b_1 + a_2b_2 + a_3b_3)\, e_3 \end{pmatrix}$$

$$= (\vec{c}\,\vec{a}^t)\,\vec{b}$$

$$\vec{a} \times \vec{b} = \begin{pmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix} = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = A\,\vec{b}$$

Donc, $\vec{r}' = \vec{e}\vec{e}^t\, r + (\cos\Theta)\,\vec{r} - (\cos\Theta)\,\vec{e}\vec{e}^t\,\vec{r} + (\sin\Theta)\,C\,\vec{r}$

où $C = \begin{pmatrix} 0 & -e_3 & 0 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{pmatrix}$

C'est-à-dire $\vec{r}' = A\,\vec{r}$

où $A = \vec{e}\vec{e}^t + (\cos\Theta) - (\cos\Theta)\,\vec{e}\vec{e}^t + (\sin\Theta)\,C$

or $\vec{e}\vec{e}^t = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} (e_1\ e_2\ e_3)$

Donc $A = D + (\cos\Theta)(I - D) + (\sin\Theta)\,C$

Les nouvelles directions des axes sont données par:

$$\vec{e}_i' = \vec{e}_i A = \begin{pmatrix} a_{1i} \\ a_{2i} \\ a_{3i} \end{pmatrix}$$

D'où les colonnes de A sont les directions des nouveaux axes en terme des anciennes coordonnées.

2) Prouvons que la matrice A est orthogonale.

Les éléments de A sont:

$$\begin{pmatrix} e_1^2 + \cos\Theta\,(1 - e_1^2) & e_1 e_2\,(1 - \cos\Theta) - e_3 \sin\Theta & e_3 e_1\,(1 - \cos\Theta) + e_2 \sin\Theta \\ e_1 e_2\,(1 - \cos\Theta) + e_3 \sin\Theta & e_2^2 + \cos\Theta\,(1 - e_2^2) & e_3 e_2\,(1 - \cos\Theta) - e_1 \sin\Theta \\ e_1 e_3\,(1 - \cos\Theta) - e_2 \sin\Theta & e_2 e_3\,(1 - \cos\Theta) + e_1 \sin\Theta & e_3^2 + \cos\Theta\,(1 - e_3^2) \end{pmatrix}$$

La matrice correspondant à une rotation $-\Theta$ est la matrice A^t.

Ainsi, si $\vec{r}' = A\vec{r}$ et $\vec{r}'' = A^t\vec{r}'$,
alors $\vec{r}'' = A^t\vec{r}' = A^tA\vec{r} = \vec{r}$.
Ceci est vrai pour tout vecteur $\vec{r}$.
D'où $A^tA = I$.
C'est-à-dire A est une matrice orthogonale.

f) Sytème d'équations

Nous montrons que la résolution du système d'équations

$$[\,(\mathrm{Tr}\,M)\,I - M\,]\begin{pmatrix}\delta\, e_1\\ \delta\, e_2\\ \delta\, e_3\end{pmatrix}=\begin{pmatrix}\Sigma_{i=1,E}\,(x_{i3}\,y_{i2} - x_{i2}\,y_{i3})\\ \Sigma_{i=1,E}\,(x_{i1}\,y_{i3} - x_{i3}\,y_{i1})\\ \Sigma_{i=1,E}\,(x_{i2}\,y_{i1} - x_{i1}\,y_{i2})\end{pmatrix}$$

$$\text{où } M=\begin{pmatrix}\Sigma_{i=1,E}\,y_{i1}^2 & \Sigma_{i=1,E}\,y_{i1}\,y_{i2} & \Sigma_{i=1,E}\,y_{i1}\,y_{i3}\\ \Sigma_{i=1,E}\,y_{i2}\,y_{i1} & \Sigma_{i=1,E}\,y_{i2}^2 & \Sigma_{i=1,E}\,y_{i2}\,y_{i3}\\ \Sigma_{i=1,E}\,y_{i3}\,y_{i1} & \Sigma_{i=1,E}\,y_{i3}\,y_{i2} & \Sigma_{i=1,E}\,y_{i3}^2\end{pmatrix}$$

$$\mathrm{Tr}\,M = \Sigma_{i=1,E}\,y_{i1}^2 + y_{i2}^2 + y_{i3}^2$$

I = matrice identité 3×3

nous donne la solution du problème:

$$\text{minimiser } F_2(\delta\, e_1, \delta\, e_2, \delta\, e_3) = \Sigma_{i=1,E}\,[\,[x_{i1} - (A'y_i)_1]^2 + [x_{i2} - (A'y_i)_2]^2 + [x_{i3} - (A'y_i)_3]^2\,] \qquad (2.7)$$

où $A' = I + \delta\, C$;

δ angle de rotation autour de l'axe de vecteur $\vec{e}$;

e_1, e_2, e_3 cosinus directeurs du vecteur $\vec{e}$;

la fonction objectif F_2 dépend de trois variables:

$\delta\, e_1, \delta\, e_2, \delta\, e_3$.

En effet

$$\text{On a } A'y_i = \begin{pmatrix} 1 & -\delta e_3 & \delta e_2 \\ \delta e_3 & 1 & -\delta e_1 \\ -\delta e_2 & \delta e_1 & 1 \end{pmatrix} \begin{pmatrix} y_{i1} \\ y_{i2} \\ y_{i3} \end{pmatrix}$$

$$= \begin{pmatrix} y_{i1} - \delta e_3 y_{i2} + \delta e_2 y_{i3} \\ \delta e_3 y_{i1} + y_{i2} - \delta e_1 y_{i3} \\ -\delta e_2 y_{i1} + \delta e_1 y_{i2} + y_{i3} \end{pmatrix}$$

Remplaçons $A'y_i$ par sa valeur dans F_2, on a:

$$F_2 = \Sigma_{i=1,E} \{ [y_{i1} - \delta e_3 y_{i2} + \delta e_2 y_{i3} - x_{i1}]^2 + [\delta e_3 y_{i1} + y_{i2} - \delta e_1 y_{i3} - x_{i2}]^2 + [- \delta e_2 y_{i1} + \delta e_1 y_{i2} + y_{i3} - x_{i3}]^2 \}$$

Développons l'expression:

$$\begin{aligned} F_2 = \Sigma_{i=1,E} \{ & y_{i1}^2 + \delta^2 e_3^2 y_{i2}^2 + \delta^2 e_2^2 y_{i3}^2 + x_{i1}^2 - 2\delta e_3 y_{i1} y_{i2} + 2\delta e_2 y_{i1} y_{i3} - 2 y_{i1} x_{i1} \\ & - 2\delta^2 e_2 e_3 y_{i2} y_{i3} + 2\delta e_3 y_{i2} x_{i1} - 2\delta e_2 y_{i3} x_{i1} \\ & + \delta^2 e_3^2 y_{i1}^2 + y_{i2}^2 + \delta^2 e_1^2 y_{i3}^2 + x_{i2}^2 + 2\delta e_3 y_{i1} y_{i2} - 2\delta^2 e_1 e_3 y_{i1} y_{i3} - 2\delta e_3 y_{i1} x_{i2} \\ & - 2\delta e_1 y_{i2} y_{i3} - 2 y_{i2} x_{i2} + 2\delta e_1 y_{i3} x_{i2} \\ & + \delta^2 e_2^2 y_{i1}^2 + \delta^2 e_1^2 y_{i2}^2 + y_{i3}^2 + x_{i3}^2 - 2\delta^2 e_1 e_2 y_{i1} y_{i2} - 2\delta e_2 y_{i1} y_{i3} + 2\delta e_2 y_{i1} x_{i3} \\ & + 2\delta e_1 y_{i2} y_{i3} - 2\delta e_1 y_{i2} x_{i3} - 2 y_{i3} x_{i3} \} \end{aligned}$$

Après simplification:

$$\begin{aligned} F_2 = \Sigma_{i=1,E} \{ & y_{i1}^2 + y_{i2}^2 + y_{i3}^2 + x_{i1}^2 + x_{i2}^2 + x_{i3}^2 + \delta^2 e_3^2 (y_{i2}^2 + y_{i1}^2) \\ & + \delta^2 e_2^2 (y_{i3}^2 + y_{i1}^2) + \delta^2 e_1^2 (y_{i3}^2 + y_{i2}^2) - 2\delta^2 (e_2 e_3 y_{i2} y_{i3} + e_1 e_3 y_{i1} y_{i3} \\ & + e_1 e_2 y_{i1} y_{i2}) + 2\delta [e_3 (y_{i2} x_{i1} - y_{i1} x_{i2}) + e_2 (y_{i1} x_{i3} - y_{i3} x_{i1}) \\ & + e_1 (y_{i3} x_{i2} - y_{i2} x_{i3})] \} \end{aligned}$$

C'est-à-dire:

$$F_2 = \Sigma_{i=1,E} \{ y_{i1}^2 + y_{i2}^2 + y_{i3}^2 + x_{i1}^2 + x_{i2}^2 + x_{i3}^2 + \delta^2[e_3^2(y_{i2}^2 + y_{i1}^2) + e_2^2(y_{i3}^2 + y_{i1}^2) + e_1^2(y_{i3}^2 + y_{i2}^2)] - 2\delta^2(e_2e_3y_{i2}y_{i3} + e_1e_3y_{i1}y_{i3} + e_1e_2y_{i1}y_{i2}) + 2\delta [e_3(y_{i2}x_{i1} - y_{i1}x_{i2}) + e_2(y_{i1}x_{i3} - y_{i3}x_{i1}) + e_1(y_{i3}x_{i2} - y_{i2}x_{i3})] \}$$

La solution du problème (2.7) est l'angle δ^* tel que:

$$\frac{\partial F_2}{\partial \delta e_1}(\delta^*) = 0, \quad \frac{\partial F_2}{\partial \delta e_2}(\delta^*) = 0, \quad \frac{\partial F_2}{\partial \delta e_3}(\delta^*) = 0 \qquad (2.8)$$

et $\nabla^2 F_2 > 0$ (2.9)

Recherchons la solution de (2.8):

$$\Sigma_{i=1,E} \{ 2\delta e_1(y_{i3}^2 + y_{i2}^2) - 2\delta(e_3y_{i1}y_{i3} + e_2y_{i1}y_{i2}) + 2(y_{i3}x_{i2} - y_{i2}x_{i3}) \} = 0$$
$$\Sigma_{i=1,E} \{ 2\delta e_2(y_{i3}^2 + y_{i1}^2) - 2\delta(e_3y_{i2}y_{i3} + e_1y_{i1}y_{i2}) + 2(y_{i1}x_{i3} - y_{i3}x_{i1}) \} = 0$$
$$\Sigma_{i=1,E} \{ 2\delta e_3(y_{i2}^2 + y_{i1}^2) - 2\delta(e_2y_{i2}y_{i3} + e_1y_{i1}y_{i3}) + 2(y_{i2}x_{i1} - y_{i1}x_{i2}) \} = 0$$

Nous obtenons le système de trois équations à trois inconnues:

$$\Sigma_{i=1,E}\ \delta e_1(y_{i3}^2 + y_{i2}^2) - \delta e_2y_{i1}y_{i2} - \delta e_3y_{i1}y_{i3} = \Sigma_{i=1,E}\ y_{i2}x_{i3} - y_{i3}x_{i2}$$
$$\Sigma_{i=1,E}\ - \delta e_1y_{i1}y_{i2} + \delta e_2(y_{i3}^2 + y_{i1}^2) - \delta e_3y_{i2}y_{i3} = \Sigma_{i=1,E}\ y_{i3}x_{i1} - y_{i1}x_{i3}$$
$$\Sigma_{i=1,E}\ - \delta e_1y_{i1}y_{i3} - \delta e_2y_{i2}y_{i3} + \delta e_3(y_{i2}^2 + y_{i1}^2) = \Sigma_{i=1,E}\ y_{i1}x_{i2} - y_{i2}x_{i1}$$

Ce sytème s'écrit matriciellement:

$$[\,(\mathrm{Tr}\,M)\,I - M\,] \begin{matrix} \delta e_1 \\ \delta e_2 \\ \delta e_3 \end{matrix} = \begin{pmatrix} \Sigma_{i=1,E}\,(\,x_{i3}\,y_{i2} - x_{i2}\,y_{i3}\,) \\ \Sigma_{i=1,E}\,(\,x_{i1}\,y_{i3} - x_{i3}\,y_{i1}\,) \\ \Sigma_{i=1,E}\,(\,x_{i2}\,y_{i1} - x_{i1}\,y_{i2}\,) \end{pmatrix}$$

$$\text{où} \quad M = \begin{pmatrix} \Sigma_{i=1,E}\,y_{i1}^{\,2} & \Sigma_{i=1,E}\,y_{i1}\,y_{i2} & \Sigma_{i=1,E}\,y_{i1}\,y_{i3} \\ \Sigma_{i=1,E}\,y_{i2}\,y_{i1} & \Sigma_{i=1,E}\,y_{i2}^{\,2} & \Sigma_{i=1,E}\,y_{i2}\,y_{i3} \\ \Sigma_{i=1,E}\,y_{i3}\,y_{i1} & \Sigma_{i=1,E}\,y_{i3}\,y_{i2} & \Sigma_{i=1,E}\,y_{i3}^{\,2} \end{pmatrix}$$

$$\mathrm{Tr}\,M = \Sigma_{i=1,E}\,y_{i1}^{\,2} + y_{i2}^{\,2} + y_{i3}^{\,2}$$

Regardons (2.9):

$$\frac{\partial^2 F_2}{\partial\,\delta e_1^{\,2}} = \Sigma_{i=1,E}\,(\,y_{i3}^{\,2} + y_{i2}^{\,2})$$

$$\frac{\partial^2 F_2}{\partial\,\delta e_2^{\,2}} = \Sigma_{i=1,E}\,(\,y_{i3}^{\,2} + y_{i1}^{\,2})$$

$$\frac{\partial^2 F_2}{\partial\,\delta e_3^{\,2}} = \Sigma_{i=1,E}\,(\,y_{i2}^{\,2} + y_{i1}^{\,2})$$

Or $E \geq 3$ et y_1 est le seul atome en l'origine.
D'où y_{i1}, y_{i2}, y_{i3}, $i=2,\ldots,E$ ne sont pas tous nuls.
Donc $\nabla^2 F > 0$.

2.4 Fitting flexible

Le fitting flexible, développé par Barino en 1980 [6] dans un programme appelé FMFIT (Flexible Molecular FIT), tient compte de la flexibilité de la molécule.

La flexibilité est conditionnée par l'existence de liaisons simples. Par contre, la présence de liaisons doubles, triples ou de cycles engendrent une rigidité de la molécule.

Cette flexibilité nous permet de modifier la configuration de la molécule. En effet, par rapport à une liaison simple, on peut décomposer la molécule en une partie mobile contenant les atomes à comparer et une partie fixe. En faisant pivoter la partie mobile autour de la liaison, nous modifions la configuration.

a) Problème

Au départ d'une liaison simple prise comme axe de rotation, nous effectuons la translation des deux molécules de façon que la liaison choisie passe par l'origine du système de référence.

Le problème consiste alors à rechercher l'angle de rotation de la partie mobile faisant concorder au mieux les atomes à comparer. Nous recherchons donc l'angle optimal minimisant la somme des distances au carré entre les éléments de AFE et AME.

Le problème s'écrit:

$$\text{minimiser } F(\delta) = \Sigma_{i=1,E} \{ [x_{i1} - (Ay_i)_1]^2 + [x_{i2} - (Ay_i)_2]^2 + [x_{i3} - (Ay_i)_3]^2 \} \qquad (2.10)$$

où A est la matrice exprimant la rotation d'un angle δ de la partie mobile de la molécule autour de la liaison simple de vecteur $\vec{e}$

$$A = (\cos\delta)\, I + (1 - \cos\delta)\, D + (\sin\delta)\, C \qquad (2.11)$$

$$D = \begin{pmatrix} e_1^2 & e_1e_2 & e_1e_3 \\ e_2e_1 & e_2^2 & e_2e_3 \\ e_3e_1 & e_3e_2 & e_3^2 \end{pmatrix}$$

$$C = \begin{pmatrix} 0 & -e_3 & 0 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{pmatrix}$$

I = matrice identité 3 x 3

L'axe de rotation étant connu, seul l'angle de rotation ne l'est pas. Le problème est donc un problème de minimisation d'une fonction non linéaire à une seule variable : δ, l'angle de rotation.

L'angle δ^* solution de (2.7) est la valeur telle que:

$$\frac{dF}{d\delta}(\delta^*) = 0 \quad \text{et} \quad \frac{d^2F}{d\delta^2}(\delta^*) > 0$$

La solution de l'équation $\frac{dF}{d\delta}(\delta) = 0$ est

$$\delta = \text{TAN}^{-1} \frac{\Sigma_{i=1,E} \{ e_3 (y_{i1} x_{i2} - y_{i2} x_{i1}) + e_2 (y_{i3} x_{i1} - y_{i1} x_{i3})}{-\Sigma_{i=1,E} \{ (e_1^2 - 1) y_{i1} x_{i1} + (e_2^2 - 1) y_{i2} x_{i2} + (e_3^2 - 1) y_{i3} x_{i3} + e_1 (y_{i2} x_{i3} - y_{i3} x_{i2}) \}}{e_1 e_2 (y_{i2} x_{i1} + y_{i1} x_{i2}) + e_1 e_3 (y_{i3} x_{i1} + y_{i1} x_{i3}) + e_3 e_2 (y_{i3} x_{i2} + y_{i2} x_{i3}) \}} \quad (2.12)$$

La solution de (2.7) est δ ou $\delta + \pi$

selon que $\frac{d^2F}{d\delta^2}(\delta) > 0$ ou $\frac{d^2F}{d\delta^2}(\delta) < 0$

En effet : - Si $\frac{d^2F}{d\delta^2}(\delta) > 0$

alors $\delta^* = \delta$ est un minimum.

- Si $\frac{d^2F}{d\delta^2}(\delta) < 0$

alors δ est un maximum.

Or la fonction $F(\delta)$ peut s'écrire comme:

$$F(\delta) = A + B \cos \delta + C \sin \delta$$

où A, B, C sont des constantes.

$F(\delta)$ est donc une fonction 2π-périodique, ayant sur une période un maximum et un minimum distants l'un de l'autre de π.

La solution est donc $\delta^* = \delta + \pi$.

b) Inconvénients de la méthode

Le fitting flexible tient compte de liaisons simples de la molécule mobile et peut donc mieux la faire coïncider avec la molécule de référence.

Cette méthode comporte néanmoins trois inconvénients:

1. Dans sa version actuelle, le programme FMFIT ne tient pas compte de notions d'énergie. Un fitting flexible peut donc conduire à une conformation moléculaire de trop haute énergie voire tout à fait inconcevable de par la proximité de groupements volumineux par exemple.

Nous développerons au chapitre 3 un fitting flexible tenant compte de notions d'énergie de conformation de la molécule.

2. Le programme actuel n'utilise qu'une seule liaison simple pour faire coïncider les atomes à comparer. Il faut donc effectuer plusieurs fittings successifs pour pouvoir faire concorder les atomes.

Nous étudierons au chapitre 4 un fitting tenant compte de plusieurs liaisons simples .

3. Le programme FMFIT ne vérifie pas que la liaison choisie comme axe de rotation est une liaison simple ne faisant pas partie d'un cycle.

De plus, le programme ne détermine pas les parties mobile et fixe de la molécule mobile. Cette tâche est donc à charge de l'utilisateur.

Nous développerons au chapitre 5 des méthodes permettant de remédier aux lacunes énoncées ci-dessus.

C) Dérivée première

Nous montrons que la solution δ de l'équation $\frac{dF}{d\delta}(\delta) = 0$ est l'expression:

$$\delta = \text{TAN}^{-1} \frac{\Sigma_{i=1,E} \{ e_3 (y_{i1}x_{i2} - y_{i2}x_{i1}) + e_2 (y_{i3}x_{i1} - y_{i1}x_{i3}) - \Sigma_{i=1,E} \{ (e_1^2-1) y_{i1}x_{i1} + (e_2^2-1) y_{i2}x_{i2} + (e_3^2-1) y_{i3}x_{i3} + e_1 (y_{i2}x_{i3} - y_{i3}x_{i2}) \}}{e_1e_2 (y_{i2}x_{i1} + y_{i1}x_{i2}) + e_1e_3 (y_{i3}x_{i1} + y_{i1}x_{i3}) + e_3e_2 (y_{i3}x_{i2} + y_{i2}x_{i3}) \}}$$

En effet

Rappelons la fonction objectif:

$$F(\delta) = \Sigma_{i=1,E} \{ [x_{i1} - (Ay_i)_1]^2 + [x_{i2} - (Ay_i)_2]^2 + [x_{i3} - (Ay_i)_3]^2 \}$$

Par l'expression de la matrice A et des coordonnées de y_i, on a:

$$Ay_i = \begin{pmatrix} \cos\delta + e_1^2(1-\cos\delta) & (1-\cos\delta)e_1e_2 - e_3\sin\delta & (1-\cos\delta)e_1e_3 + e_2\sin\delta \\ (1-\cos\delta)e_2e_1 + e_3\sin\delta & \cos\delta + e_2^2(1-\cos\delta) & (1-\cos\delta)e_2e_3 - e_1\sin\delta \\ (1-\cos\delta)e_3e_1 - e_2\sin\delta & (1-\cos\delta)e_3e_2 + e_1\sin\delta & \cos\delta + e_3^2(1-\cos\delta) \end{pmatrix} \begin{pmatrix} y_{i1} \\ y_{i2} \\ y_{i3} \end{pmatrix}$$

$$= \begin{pmatrix} \{ \cos\delta + e_1^2(1-\cos\delta) \} y_{i1} + \{ (1-\cos\delta)e_1e_2 - e_3\sin\delta \} y_{i2} + \{ (1-\cos\delta)e_1e_3 + e_2\sin\delta \} y_{i3} \\ \{ (1-\cos\delta)e_2e_1 + e_3\sin\delta \} y_{i1} + \{ \cos\delta + e_2^2(1-\cos\delta) \} y_{i2} + \{ (1-\cos\delta)e_2e_3 - e_1\sin\delta \} y_{i3} \\ \{ (1-\cos\delta)e_3e_1 - e_2\sin\delta \} y_{i1} + \{ (1-\cos\delta)e_3e_2 + e_1\sin\delta \} y_{i2} + \{ \cos\delta + e_3^2(1-\cos\delta) \} y_{i3} \end{pmatrix}$$

Remplaçons Ay_i par sa valeur dans la fonction F (δ) et développons:

$$
\begin{aligned}
F(\delta) = \Sigma_{i=1,E} \{ & [\cos\delta + e_1^2(1-\cos\delta)]^2 y_{i1}^2 + [(1-\cos\delta)e_1e_2 - e_3\sin\delta]^2 y_{i2}^2 \\
& + [(1-\cos\delta)e_1e_3 + e_2\sin\delta]^2 y_{i3}^2 + x_{i1}^2 + 2[\cos\delta + e_1^2(1-\cos\delta)]y_{i1} \\
& [(1-\cos\delta)e_1e_2 - e_3\sin\delta]y_{i2} + 2[\cos\delta + e_1^2(1-\cos\delta)]y_{i1} \\
& [(1-\cos\delta)e_1e_3 + e_2\sin\delta]y_{i3} - 2[\cos\delta + e_1^2(1-\cos\delta)]y_{i1}x_{i1} \\
& + 2[(1-\cos\delta)e_1e_2 - e_3\sin\delta]y_{i2}\,[(1-\cos\delta)e_1e_3 + e_2\sin\delta]y_{i3} \\
& - 2[(1-\cos\delta)e_1e_2 - e_3\sin\delta]y_{i2}x_{i1} - 2[(1-\cos\delta)e_1e_3 + e_2\sin\delta]y_{i3}x_{i1} \\
& + [(1-\cos\delta)e_1e_2 + e_3\sin\delta]^2 y_{i1}^2 + [\cos\delta + e_2^2(1-\cos\delta)]^2 y_{i2}^2 \\
& + [(1-\cos\delta)e_2e_3 - e_1\sin\delta]^2 y_{i3}^2 + x_{i2}^2 + 2[(1-\cos\delta)e_1e_2 + \\
& e_3\sin\delta]y_{i1}[\cos\delta + e_2^2(1-\cos\delta)]y_{i2} + 2\,[(1-\cos\delta)e_1e_2 + e_3\sin\delta]y_{i1} \\
& [(1-\cos\delta)e_2e_3 - e_1\sin\delta]y_{i3} - 2[(1-\cos\delta)e_1e_2 + e_3\sin\delta]y_{i1}x_{i2} \\
& + 2[\cos\delta + e_2^2(1-\cos\delta)]y_{i2}[(1-\cos\delta)e_2e_3 - e_1\sin\delta]y_{i3} \\
& - 2[\cos\delta + e_2^2(1-\cos\delta)]y_{i2}x_{i2} - 2[(1-\cos\delta)e_2e_3 - e_1\sin\delta]y_{i3}x_{i2} \\
& + [(1-\cos\delta)e_3e_1 - e_2\sin\delta]^2 y_{i1}^2 + [(1-\cos\delta)e_3e_2 + e_1\sin\delta]^2 y_{i2}^2 \\
& + [\cos\delta + e_3^2(1-\cos\delta)]^2 y_{i3}^2 + x_{i3}^2 + 2\,[(1-\cos\delta)e_3e_1 - e_2\sin\delta]y_{i1} \\
& [(1-\cos\delta)e_3e_2 + e_1\sin\delta]y_{i2} + 2\,[(1-\cos\delta)e_3e_1 - e_2\sin\delta]y_{i1} \\
& [\cos\delta + e_3^2(1-\cos\delta)]y_{i3} - 2[(1-\cos\delta)e_3e_1 - e_2\sin\delta]y_{i1}x_{i3} \\
& + 2\,[(1-\cos\delta)e_3e_2 + e_1\sin\delta]y_{i2}[\cos\delta + e_3^2(1-\cos\delta)]y_{i3} \\
& - 2[(1-\cos\delta)e_2e_3 + e_1\sin\delta]y_{i2}x_{i3} - 2[\cos\delta + e_3^2(1-\cos\delta)]y_{i3}x_{i3} \}
\end{aligned}
$$

Développons:

$$
\begin{aligned}
F(\delta) = \Sigma_{i=1,E} \{ & [\cos^2\delta + e_1^4\cos^2\delta + e_1^4 - 2e_1^4\cos\delta + 2e_1^2\cos\delta - 2e_1^2\cos^2\delta]y_{i1}^2 \\
& + [e_1^2e_2^2 + e_1^2e_2^2\cos^2\delta - 2e_1^2e_2^2\cos\delta + e_3^2\sin^2\delta - 2e_1e_2e_3\sin\delta \\
& \qquad + 2e_1e_2e_3\sin\delta\cos\delta]y_{i2}^2 \\
& + [e_1^2e_3^2 + e_1^2e_3^2\cos^2\delta - 2e_1^2e_3^2\cos\delta + e_2^2\sin\delta + 2e_1e_2e_3\sin\delta \\
& \qquad - 2e_1e_2e_3\sin\delta\cos\delta]\,y_{i3}^2 + x_{i1}^2 \\
& + 2y_{i1}y_{i2}[e_1e_2\cos\delta - e_1e_2\cos^2\delta - e_3\sin\delta\cos\delta + e_1^3e_2 - 2e_1^2e_2\cos\delta \\
& \qquad + e_1^3e_2\cos^2\delta - e_1^2e_3\sin\delta + e_1^2e_3\sin\delta\cos\delta] \\
& + 2y_{i1}y_{i3}[\,e_1e_3\cos\delta - e_1e_3\cos^2\delta + e_2\sin\delta\cos\delta + e_1^3e_3 - 2e_1^3e_3\cos\delta \\
& \qquad + e_1^3e_3\cos^2\delta + e_1^2e_2\sin\delta - e_1^2e_3\sin\delta\cos\delta]
\end{aligned}
$$

$$
\begin{aligned}
&- 2y_{i1}x_{i1}[\cos\delta + e_1^2 - e_1^2\cos\delta] \\
&+ 2y_{i2}y_{i3}[e_1^2e_2e_3 - 2e_1^2e_2e_3\cos\delta + e_1^2e_2e_3\cos^2\delta + e_1e_2^2\sin\delta \\
&\qquad - e_1e_2^2\sin\delta\cos\delta - e_1e_3^2\sin\delta + e_1e_3^2\sin\delta\cos\delta \\
&\qquad - e_2e_3\sin^2\delta] \\
&- 2y_{i2}x_{i1}[e_1e_2 - e_1e_2\cos\delta - e_3\sin\delta] \\
&- 2y_{i3}x_{i1}[e_1e_3 - e_1e_3\cos\delta + e_2\sin\delta] \\
&+ [e_1^2e_2^2 + e_1^2e_2^2\cos^2\delta - 2e_1^2e_2^2\cos\delta + e_3^2\sin^2\delta + 2e_1e_2e_3\sin\delta \\
&\qquad - 2e_1e_2e_3\sin\delta\cos\delta]y_{i1}^2 \\
&+ [\cos^2\delta + e_2^4\cos^2\delta + e_2^4 - 2e_2^4\cos\delta + 2e_2^2\cos\delta - 2e_2^2\cos^2\delta]y_{i2}^2 \\
&+ [e_2^2e_3^2 + e_2^2e_3^2\cos^2\delta - 2e_2^2e_3^2\cos\delta + e_1^2\sin^2\delta - 2e_1e_2e_3\sin\delta \\
&\qquad + 2e_1e_2e_3\sin\delta\cos\delta]y_{i3}^2 + x_{i2}^2 \\
&+ 2y_{i1}y_{i2}[e_1e_2\cos\delta - e_1e_2\cos^2\delta + e_1e_2^3 - 2e_1e_2^3\cos\delta + e_1e_2^3\cos^2\delta \\
&\qquad + e_3\sin\delta\cos\delta + e_2^2e_3\sin\delta - e_2^2e_3\sin\delta\cos\delta] \\
&+ 2y_{i1}y_{i3}[e_1e_2^2e_3 - 2e_1e_2^2e_3\cos\delta + e_1e_2^2e_3\cos^2\delta - e_1^2e_3\sin\delta \\
&\qquad + e_1^2e_2\sin\delta\cos\delta + e_2e_3^2\sin\delta - e_2e_3^2\sin\delta\cos\delta \\
&\qquad - e_1e_3\sin^2\delta] \\
&- 2y_{i1}x_{i2}[e_1e_2 - e_1e_2\cos\delta + e_3\sin\delta] \\
&+ 2y_{i2}y_{i3}[e_2e_3\cos\delta - e_2e_3\cos^2\delta - e_1\sin\delta\cos\delta + e_2^3e_3 - 2e_2^3e_3\cos\delta \\
&\qquad + e_2^3e_3\cos^2\delta - e_1e_2^2\sin\delta + e_1e_2^2\sin\delta\cos\delta] \\
&- 2y_{i2}x_{i2}[\cos\delta + e_2^2 - e_2^2\cos\delta] - 2y_{i3}x_{i2}[e_2e_3 - e_2e_3\cos\delta - e_1\sin\delta] \\
&+ [e_1^2e_3^2 + e_1^2e_3^2\cos^2\delta - 2e_1^2e_3^2\cos\delta + e_2^2\sin^2\delta - 2e_1e_2e_3\sin\delta \\
&\qquad + 2e_1e_2e_3\sin\delta\cos\delta]y_{i1}^2 \\
&+ [e_2^2e_3^2 + e_2^2e_3^2\cos^2\delta - 2e_2^2e_3^2\cos\delta + e_1^2\sin^2\delta + 2e_1e_2e_3\sin\delta \\
&\qquad - 2e_1e_2e_3\sin\delta\cos\delta]y_{i2}^2 \\
&+ [\cos^2\delta + e_3^4\cos^2\delta + e_3^4 - 2e_3^4\cos\delta + 2e_3^2\cos\delta - 2e_3^2\cos^2\delta]y_{i3}^2 \\
&+ x_{i3}^2 \\
&+ 2y_{i1}y_{i2}[e_1e_2e_3^2 - 2e_1e_2e_3^2\cos\delta + e_1e_2e_3^2\cos^2\delta + e_1^2e_3\sin\delta \\
&\qquad - e_1^2e_3\sin\delta\cos\delta - e_3e_2^2\sin\delta + e_2e_3^2\sin\delta\cos\delta \\
&\qquad - e_1e_2\sin^2\delta] \\
&+ 2y_{i1}y_{i3}[e_1e_3^3 - 2e_1e_3^3\cos\delta + e_1e_3^3\cos^2\delta + e_3e_1\cos\delta - e_1e_3\cos^2\delta \\
&\qquad - e_3^2e_2\sin\delta + e_3^2e_2\sin\delta\cos\delta - e_2\sin\delta\cos\delta] \\
&- 2y_{i1}x_{i3}[e_1e_3 - e_1e_3\cos\delta - e_2\sin\delta]
\end{aligned}
$$

$$+ 2y_{i2}y_{i3}[e_2e_3^3 - 2e_2e_3^3\cos\delta + e_2e_3^3\cos^2\delta + e_2e_3\cos\delta - e_2e_3\cos^2\delta + e_1e_3^2\sin\delta - e_2e_3^2\sin\delta\cos\delta + e_1\sin\delta\cos\delta]$$
$$- 2y_{i2}x_{i3}[e_2e_3 - e_2e_3\cos\delta + e_1\sin\delta] - 2y_{i3}x_{i3}[\cos\delta + e_3^2 - e_3^2\cos\delta]$$

Regardons les termes en y_{i1}^2:

$$\cos^2\delta\,(1 + e_1^4 - 2e_1^2 + e_1^2e_2^2 + e_1^2e_3^2) + \cos\delta\,(-2e_1^4 + 2e_1^2 - 2e_1^2e_2^2 - 2e_1^2e_3^2) + \sin^2\delta\,(e_3^2 + e_2^2) + (e_1^4 + e_1^2e_2^2 + e_1^2e_3^2)]$$
$$= \cos^2\delta\,[1 + e_1^4 - 2e_1^2 + e_1^2(e_2^2 + e_3^2)] + \cos\delta\,(-2e_1^4 + 2e_1^2 - 2e_1^2(1 - e_2^2)] + \sin^2\delta\,(1 - e_1^2) + [e_1^4 + e_1^2(e_2^2 + e_3^2)]$$
$$= \cos^2\delta\,[1 - e_1^2 + e_1^4 - e_1^2 + e_1^2(1 - e_1^2)] + \cos\delta\,(-2e_1^4 + 2e_1^2 - 2e_1^2 + 2e_2^4) + \sin^2\delta\,(1 - e_1^2) + [e_1^4 + e_1^2(1 - e_1^2)]$$
$$= \cos^2\delta\,(1 - e_1^2) + \sin^2\delta\,(1 - e_1^2) + e_1^2$$
$$= 1$$

De même, la somme des termes en y_{i2}^2 et y_{i3}^2 valent 1.
Introduisons ces résultats dans F (δ) et après simplification, on a:

$$F(\delta) = \Sigma_{i=1,E}\{ y_{i1}^2 + y_{i2}^2 + y_{i3}^2 + x_{i1}^2 + x_{i2}^2 + x_{i3}^2$$
$$+ 2y_{i1}y_{i2}[e_1e_2\cos\delta - e_1e_2\cos^2\delta + e_1^3e_2 - 2e_1^3e_2\cos\delta + e_1^3e_2\cos^2\delta - e_1^2e_3\sin\delta]$$
$$+ 2y_{i1}y_{i3}[e_1e_3\cos\delta - e_1e_3\cos^2\delta + e_1^3e_3 - 2e_1^3e_3\cos\delta + e_1^3e_3\cos^2\delta + e_1^2e_2\sin\delta]$$
$$- 2y_{i1}x_{i1}[\cos\delta + e_1^2 - e_1^2\cos\delta]$$
$$+ 2y_{i2}y_{i3}[e_1^2e_2e_3 - 2e_1^2e_2e_3\cos\delta + e_1^2e_2e_3\cos^2\delta + e_1e_2^2\sin\delta - e_1e_3^2\sin\delta - e_2e_3\sin^2\delta]$$
$$- 2y_{i2}x_{i1}[e_1e_2 - e_1e_2\cos\delta - e_3\sin\delta]$$
$$- 2y_{i3}x_{i1}[e_1e_3 - e_1e_3\cos\delta + e_2\sin\delta]$$
$$+ 2y_{i1}y_{i2}[e_1e_2\cos\delta - e_1e_2\cos^2\delta + e_1e_2^3 - 2e_1e_2^3\cos\delta + e_1e_2^3\cos^2\delta + e_2^2e_3\sin\delta]$$
$$+ 2y_{i1}y_{i3}[e_1e_2^2e_3 - 2e_1e_2^2e_3\cos\delta + e_1e_2^2e_3\cos^2\delta - e_1^2e_3\sin\delta + e_2e_3^2\sin\delta - e_1e_3\sin^2\delta]$$
$$- 2y_{i1}x_{i2}[e_1e_2 - e_1e_2\cos\delta + e_3\sin\delta]$$

$$+ 2y_{i2}y_{i3}[e_2e_3\cos\delta - e_2e_3\cos^2\delta + e_2^3e_3 - 2e_2^3e_3\cos\delta + e_2^3e_3\cos^2\delta - e_1e_2^2\sin\delta]$$
$$- 2y_{i2}x_{i2}[\cos\delta + e_2^2 - e_2^2\cos\delta] - 2y_{i3}x_{i2}[e_2e_3 - e_2e_3\cos\delta - e_1\sin\delta]$$
$$+ 2y_{i1}y_2[e_1e_2e_3^2 - 2e_1e_2e_3^2\cos\delta + e_1e_2e_3^2\cos^2\delta + e_1^2e_3\sin\delta - e_3e_2^2\sin\delta - e_1e_2\sin^2\delta]$$
$$+ 2y_{i1}y_{i3}[e_1e_3^3 - 2e_1e_3^3\cos\delta + e_1e_3^3\cos^2\delta + e_3e_1\cos\delta - e_1e_3\cos^2\delta - e_3^2e_2\sin\delta]$$
$$- 2y_{i1}x_{i3}[e_1e_3 - e_1e_3\cos\delta - e_2\sin\delta]$$
$$+ 2y_{i2}y_{i3}[e_2e_3^3 - 2e_2e_3^3\cos\delta + e_2e_3^3\cos^2\delta + e_2e_3\cos\delta - e_2e_3\cos^2\delta + e_1e_3^2\sin\delta]$$
$$- 2y_{i2}x_{i3}[e_2e_3 - e_2e_3\cos\delta + e_1\sin\delta] - 2y_{i3}x_{i3}[\cos\delta + e_3^2 - e_3^2\cos\delta]$$

Regardons les termes en $y_{i1}y_{i2}$:

$$\begin{aligned}
&2e_1e_2\cos\delta - 2e_1e_2\cos^2\delta + e_1e_2(e_1^2 + e_2^2) - 2e_1e_2\cos\delta\,(e_1^2 + e_2^2)\\
&\quad + e_1e_2\cos^2\delta\,(e_1^2 + e_2^2) - e_1e_2\sin^2\delta + e_1e_2e_3^2 - 2e_1e_2e_3^2\cos\delta\\
&\quad + e_1e_2e_3^2\cos^2\delta\\
&= 2e_1e_2e_3^2\cos\delta - 2e_1e_2e_3^2\cos\delta - e_1e_2\cos^2\delta - e_1e_2 + e_1e_2(e_1^2 + e_2^2)\\
&\quad + e_1e_2\cos^2\delta\,(e_1^2 + e_2^2) + e_1e_2e_3^2 + e_1e_2e_3^2\cos^2\delta\\
&= 0
\end{aligned}$$

De même, la somme des termes en $y_{i1}y_{i3}$ et $y_{i2}y_{i3}$ valent 0.
Introduisons ces résultats dans F (δ):

$$\begin{aligned}
F(\delta) = \Sigma_{i=1,E}\{\ & y_{i1}^2 + y_{i2}^2 + y_{i3}^2 + x_{i1}^2 + x_{i2}^2 + x_{i3}^2\\
& - 2y_{i1}x_{i1}e_1^2 - 2y_{i2}x_{i1}e_1e_2 - 2y_{i3}x_{i1}e_1e_3 - 2y_{i1}x_{i2}e_1e_2 - 2y_{i2}x_{i2}e_2^2\\
& \quad - 2y_{i3}x_{i2}e_2e_3 - 2y_{i1}x_{i3}e_1e_3 - 2y_{i2}x_{i3}e_2e_3 - 2y_{i3}x_{i3}e_3^2\\
& - 2\cos\delta\,(y_{i1}x_{i1} + y_{i2}x_{i2} + y_{i3}x_{i3}) + 2e_1^2y_{i1}x_{i1}\cos\delta + 2e_2^2y_{i2}x_{i2}\cos\delta\\
& \quad + 2e_3^2y_{i3}x_{i3}\cos\delta + 2e_1e_2y_{i2}x_{i1}\cos\delta + 2e_1e_3y_{i3}x_{i1}\cos\delta\\
& \quad + 2e_1e_2y_{i1}x_{i2}\cos\delta + 2e_2e_3y_{i3}x_{i2}\cos\delta + 2e_1e_3y_{i1}x_{i3}\cos\delta\\
& \quad + 2e_2e_3y_{i2}x_{i3}\cos\delta\\
& + 2e_3y_{i2}x_{i1}\sin\delta - 2e_2y_{i3}x_{i1}\sin\delta - 2e_3y_{i1}x_{i2}\sin\delta + 2e_1y_{i3}x_{i2}\sin\delta\\
& \quad + 2e_2y_{i1}x_{i3}\sin\delta - 2e_1y_{i2}x_{i3}\sin\delta\ \}
\end{aligned}$$

Réécrivons F (δ):

$$
\begin{aligned}
F(\delta) = \Sigma_{i=1,E} \{ & y_{i1}^2 + y_{i2}^2 + y_{i3}^2 + x_{i1}^2 + x_{i2}^2 + x_{i3}^2 \\
& - 2y_{i1}x_{i1}e_1^2 - 2y_{i2}x_{i1}e_1e_2 - 2y_{i3}x_{i1}e_1e_3 - 2y_{i1}x_{i2}e_1e_2 - 2y_{i2}x_{i2}e_2^2 \\
& \qquad - 2y_{i3}x_{i2}e_2e_3 - 2y_{i1}x_{i3}e_1e_3 - 2y_{i2}x_{i3}e_2e_3 - 2y_{i3}x_{i3}e_3^2 \\
& - 2\cos\delta\,[(e_1^2 - 1)y_{i1}x_{i1} + (e_2^2 - 1)y_{i2}x_{i2} + (e_3^2 - 1)y_{i3}x_{i3}] \\
& + 2\cos\delta\,[e_1e_2(y_{i2}x_{i1} + x_{i2}y_{i1}) + e_1e_3(y_{i3}x_{i1} + x_{i3}y_{i1}) \\
& \qquad + e_2e_3(y_{i3}x_{i2} + x_{i3}y_{i2})] \\
& + 2\sin\delta\,[e_3(y_{i2}x_{i1} - x_{i2}y_{i1}) + e_2(y_{i1}x_{i3} - x_{i1}y_{i3}) + e_1(y_{i3}x_{i2} - x_{i3}y_{i2})]
\end{aligned}
$$

Nous recherchons δ solution de l'équation $\frac{dF}{d\delta}(\delta) = 0$.

Cette équation est équivalente à:

$$
\begin{aligned}
\Sigma_{i=1,E} \{ & 2\sin\delta\,[(1 - e_1^2)y_{i1}x_{i1} + (1 - e_2^2)y_{i2}x_{i2} + (1 - e_3^2)y_{i3}x_{i3}] \\
& - 2\sin\delta\,[e_1e_2(y_{i2}x_{i1} + x_{i2}y_{i1}) + e_1e_3(y_{i3}x_{i1} + x_{i3}y_{i1}) \\
& \qquad + e_2e_3(y_{i3}x_{i2} + x_{i3}y_{i2})] \\
& + 2\cos\delta\,[e_3(y_{i2}x_{i1} - x_{i2}y_{i1}) + e_2(y_{i1}x_{i3} - x_{i1}y_{i3}) + e_1(y_{i3}x_{i2} - x_{i3}y_{i2})] \} = 0
\end{aligned}
$$

C'est-à-dire:

$$
\begin{aligned}
\cos\delta \{ & \Sigma_{i=1,E} [e_3(y_{i2}x_{i1} - x_{i2}y_{i1}) + e_2(y_{i1}x_{i3} - x_{i1}y_{i3}) + e_1(y_{i3}x_{i2} - x_{i3}y_{i2})] \} \\
= \sin\delta \{ & \Sigma_{i=1,E} [(e_1^2 - 1)y_{i1}x_{i1} + (e_2^2 - 1)y_{i2}x_{i2} + (e_3^2 - 1)y_{i3}x_{i3} \\
& + e_1e_2(y_{i2}x_{i1} + x_{i2}y_{i1}) + e_1e_3(y_{i3}x_{i1} + x_{i3}y_{i1}) + e_2e_3(y_{i3}x_{i2} + x_{i3}y_{i2})] \}
\end{aligned}
$$

Nous avons donc:

$$
\delta = \mathrm{TAN}^{-1} \frac{\Sigma_{i=1,E} \{ e_3 (y_{i1} x_{i2} - y_{i2} x_{i1}) + e_2 (y_{i3} x_{i1} - y_{i1} x_{i3}) }{ - \Sigma_{i=1,E} \{ (e_1^2 - 1) y_{i1} x_{i1} + (e_2^2 - 1) y_{i2} x_{i2} + (e_3^2 - 1) y_{i3} x_{i3} + e_1 (y_{i2} x_{i3} - y_{i3} x_{i2}) \} }{ e_1 e_2 (y_{i2} x_{i1} + y_{i1} x_{i2}) + e_1 e_3 (y_{i3} x_{i1} + y_{i1} x_{i3}) + e_3 e_2 (y_{i3} x_{i2} + y_{i2} x_{i3}) \} }
$$

Chapitre III

Fitting et énergie

Dans ce chapitre, nous tenons compte de l'énergie interne de la molécule.

Nous développons trois méthodes qui envisagent ce problème sous différents aspects.

Introduction

La théorie du fitting flexible développée au chapitre 2.3 ne tient pas compte de l'énergie de conformation de la molécule mobile.

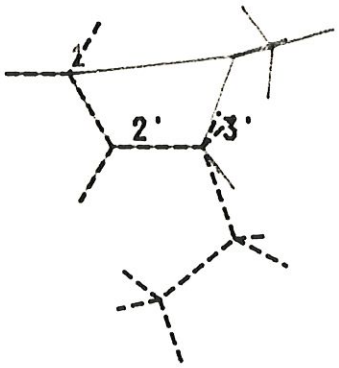
En effectuant une rotation de la partie mobile de la molécule mobile, nous modifions l'angle de torsion défini par la liaison simple servant d'axe de rotation, ainsi que certaines distances interatomiques. Nous obtenons donc une nouvelle conformation de la molécule à laquelle est associée une variation de son énergie interne basée sur la cohésion atomique, c'est-à-dire sur la force des liaisons. Il est généralement admis qu'une molécule ne modifie sa conformation que si la variation d'énergie ne dépasse pas un seuil raisonnable, de l'ordre d'une vingtaine de kcal/mole, à température ambiante.

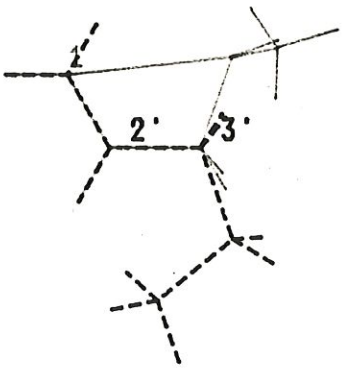
Il est possible que le résultat proposé par un fitting basé sur une stricte concordance d'atomes ne soit pas favorable énergétiquement. En effet, la conformation finale générée par le fitting présente des groupements (dans cette même molécule) trop proches l'un de l'autre. Et même si la conformation finale se trouve dans un creux au niveau énergie, celle-ci n'est pas réalisable du fait que le chemin énergétique de la conformation de départ vers la conformation finale passe par un maximum d'énergie trop supérieur aux conformations initiales et finales et infranchissables sans apport extérieur.

Exemple : Pour illustrer ceci, nous comparons les molécules méthyl-3 cyclobutene, prise pour référence, et 1-pentene, prise comme molécule à comparer.

Dans une première étape, nous effectuons un fitting rigide superposant les atomes de carbone 2, 3, 4 sur les atomes de carbone 2', 3', 4'.

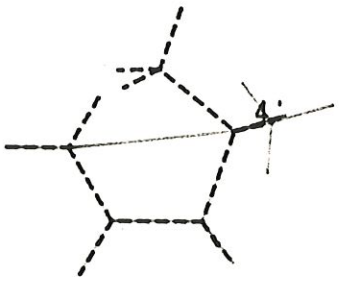
Le résultat du fitting donne une superposition exacte. Ceci n'est qu'un cas de figure car la longueur des liaisons 2-3 et 2'-3' sont égales, de même que la longueur des liaisons 3-4 et 3'-4'.

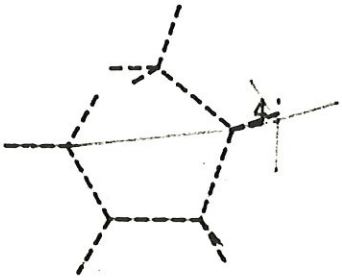




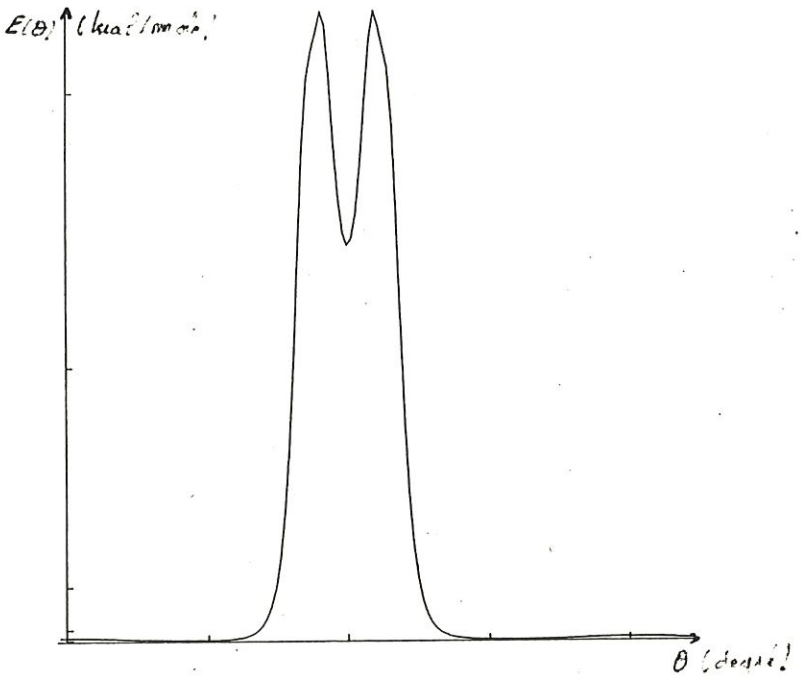
Appliquons un fitting flexible superposant l'atome de carbone 5 sur l'atome de carbone 5'.

Celui-ci superpose exactement les deux atomes et fournit un angle de 180° comme angle optimal de rotation autour de la liaison 2, 3, 4, 5.





Sans expliquer à ce stade comment cette énergie est calculée, regardons la courbe d'énergie en fonction de l'angle de rotation.



Nous voyons sur ce graphique que l'énergie de conformation obtenue par une rotation de 180° a une énergie trop élevée, 700 kcal/mole, par rapport au point de départ. De plus, le chemin énergétique pour y arriver passe par un maximum de 1150 kcal/mole.

Nous allons donc développer trois méthodes minimisant la somme des distances au carré entre les atomes à superposer mais tenant compte de l'énergie de conformation de la molécule mobile.

3.1 Energie

L'énergie de conformation d'une molécule (énergie interne à la molécule pour chaque conformation) peut être calculée par différentes méthodes théoriques telles que les méthodes non empiriques ou semi-empiriques basées sur la mécanique quantique ou des méthodes empiriques souvent basées sur la mécanique moléculaire.

Si le nombre de variables à considérer est important et s'il existe des contraintes au niveau coût et temps de calcul (par l'exemple dans le cas précis où l'application doit être couplée à un système de représentation graphique moléculaire interactif), seules des méthodes basées sur la mécanique moléculaire sont envisageables pour donner une idée de la ou des géométries préférentielles, c'est-à-dire les plus basses au niveau énergie.

Dans ce cas précis, les aspects temps et optimisation étant les plus importants, nous nous sommes basés sur la mécanique moléculaire.

En mécanique moléculaire, on considère les atomes comme des sphères reliées par des ressorts (les liaisons). On évalue donc une énergie de contrainte par une minimisation par rapport à une série de paramètres structuraux (distance, angle de valence, angle de torsion) standards, c'est-à-dire observés sur une grande quantité de composés homologues.

Les principaux types d'énergie de contrainte sont:

énergie de Van der Waals: énergie d'interaction entre atomes non liés de la molécule;
énergie de torsion: énergie qui intervient lors d'une modification de l'angle de torsion;
énergie de Coulomb: énergie d'interaction électrostatique entre atomes non liés;
énergie de "stretching": énergie de déformation de la distance entre 2 atomes liés;
énergie de "bending": énergie de déformation de l'angle de valence entre 3 atomes liés;
énergie "out of plane bending": énergie de déformation hors du plan.

Nous allons, pour calculer l'énergie de conformation, considérer les trois premiers types d'énergie, les autres types étant négligeables lors de changements de conformation par rotation autour de liaisons simples.

Examinons ces 3 types d'énergie.

Chaque type d'énergie s'exprime par des paramètres empiriques (fournis par une table) et des paramètres propres à la conformation de la molécule (paramètres qui varient lors d'une rotation autour d'une liaison simple).

L'énergie de Van der Waals, E_{VDW}, est une énergie d'interaction entre atomes non liés de la molécule.

$$E_{VDW} = \sum_{i=1,B} \sum_{j=i+1,B} f\, e^{-g r_{ij}} - h / r_{ij}^{6}$$

où f, g, h sont des paramètres empiriques;
r_{ij} est la distance entre 2 atomes non liés;
B est le nombre d'atomes de la molécule mobile.

Nous ne considérons une énergie de Van der Waals qu'entre atomes dont la distance est inférieure à 10 Å et distants l'un de l'autre d'au moins trois liaisons.

L'énergie de torsion , E_{TOR}, intervient lorsque, lors de la rotation autour d'une liaison, nous modifions l'angle de torsion.

$$E_{TOR} = \sum_{\text{angles de torsions}} \lambda_{tor} (1 \pm \cos (n_{tor} \Phi_{tor}))$$

où λ_{tor} est un paramètre empirique;
n_{tor} période de la liaison;
Φ_{tor} angle de torsion.

Nous ne considérons une énergie de torsion que pour les angles de torsion affectés par la rotation.

L'énergie de Coulomb, E_{COUL}, est une énergie d'interaction électrostatique entre atomes non liés.

$$E_{COUL} = \sum_{i=1,B} \sum_{j=i+1,B} k \frac{q_i q_j}{D r_{ij}}$$

où k est un facteur d'échelle;
D paramètre empirique;
q_i charge de l'atome i;
r_{ij} distance entre les atomes i et j;
B est le nombre d'atomes de la molécule mobile.

Nous ne considérerons une énergie de Coulomb qu'entre atomes dont la distance est inférieure à 10 Å et distants l'un de l'autre d'au moins trois liaisons.

3.2 Première approche

Dans un premier temps, nous allons étudier une méthode permettant de trouver un angle de rotation (pour la molécule mobile) minimisant la somme des distances au carré entre les atomes à comparer de chaque molécule et pour lequel l'énergie de la molécule reste inférieure ou égale au seuil défini par l'énergie minimum (appelée ENEMIN) à laquelle vient s'ajouter une tolérance (appelée TOL).

a) Recherche de l'énergie minimum

Pour une molécule comprenant m atomes, l'énergie de contrainte s'écrit :

$$E = E_{VDW} + E_{COUL} + E_{TOR} \qquad (3.1)$$

$$\text{où } E_{VDW} = \Sigma_{(i,j) \in I} \; \zeta_{ij} \left\{ 8.28\, 10^5 \exp\left(\frac{-1}{0.0736} \right) \right.$$

$$\left. \frac{r_{ij}}{r_i + r_j} \right) \; - 2.25 \left(\frac{r_i + r_j}{r_{ij}} \right)^6 \Bigg\}$$

$$E_{COUL} = \Sigma_{(i,j) \in I} \; 332.17 \; \frac{q_i \, q_j}{r_{ij}}$$

$$E_{TOR} = \Sigma_{tor \in J} \; v_{tor}^2/2 \; (1 + S_{tor} \cos (n_{tor} \Phi_{tor}))$$

où les paramètres empiriques sont:

- ζ_{ij} : constante de Van der Waals;
- r_i : rayon de Van der Waals de l'atome i;
- v_{tor} : barrière de torsion;
- n_{tor} : période de la liaison;
- S_{tor} : signe de n_{tor};

les paramètres propres à la molécule sont:
r_{ij} : distance entre les atomes i et j;
q_i : charge de l'atome i;
Φ_{tor} : angle de torsion.

$I = \{ (i,j)$ tq $1 \leq i \leq m, i+1 \leq j \leq m, r_{ij} \leq 10$ Å, les atomes i et j sont distants d'au moins trois liaisons $\}$

$J = \{\Phi_{tor}$ tq Φ_{tor} est un angle de torsion affecté par la rotation $\}$

Nous calculons le minimum d'énergie, c'est-à-dire nous résolvons le problème:

minimiser $E(\delta)$
sous $0 \leq \delta \leq 2\pi$ (3.2)

C'est un problème de minimisation d'une fonction à une variable (l'angle de rotation) non linéaire, dérivable et 2π-périodique. C'est pourquoi nous recherchons le minimum sur une période.

En effet

1. Fonction non linéaire
   L'énergie comporte une fonction cosinus qui est une fonction non linéaire.

2. Fonction dérivable (sauf pour r=0)
   L'énergie est une somme des fonctions: exponentielle, $1/r^6$, $1/r$, cosinus, constante . Toutes ces fonctions sont dérivables.

3. Fonction à une variable

Les deux seuls paramètres affectés par la rotation autour d'une liaison simple sont: la distance r_{ij} entre les atomes i, j et les angles de torsion Φ_{tor}. Ils dépendent tous deux de l'angle de rotation δ.

La fonction énergie $E(\delta)$ ne dépend donc que d'une seule variable δ.

4. Fonction 2π-périodique

Montrons que $E(\delta) = E(\delta + 2\pi)$ $\forall$ angle δ

Soit un angle δ

i) $E_{TOR}(\delta) = E_{TOR}(\delta + 2\pi)$

car les fonctions cosinus et constante sont 2π-périodiques.

ii) $E_{VDW}(\delta) + E_{COUL}(\delta) = E_{VDW}(\delta + 2\pi) + E_{COUL}(\delta + 2\pi)$

car r_{ij} est 2π-périodique

Montrons que r_{ij} est 2π-périodique.

- Si les atomes i et j sont tous deux dans la partie fixe, leur distance est constante.
- Si les atomes i et j sont tous deux dans la partie mobile, leur distance est constante car la matrice de rotation est orthogonale.
- Supposons que l'atome i $\in$ partie fixe et l'atome j $\in$ partie mobile. On a l'expression:

$$r_{ij} = \sum_{i=1,E} \{ [y_{i1} - (Ay_j)_1]^2 + [y_{i2} - (Ay_j)_2]^2 + [y_{i3} - (Ay_j)_3]^2 \}^{1/2}$$

où A est la matrice de rotation orthogonale

$$= (\cos\delta)\, I + (1 - \cos\delta)\, D + (\sin\delta)\, C$$

Nous avons déja développé une expression analogue et obtenu:

$$r_{ij} = (A + B\cos\delta + C\sin\delta)^{1/2}$$

où A, B, C sont constants

Or les fonctions $\cos\delta$ et $\sin\delta$ sont 2π-périodiques

D'où r_{ij} est 2π-périodique.

Nous allons, pour résoudre le problème du minimum d'énergie (3.2), utiliser la routine VF02AD de la librairie Harwell [7]. Cette routine minimise une fonction de plusieurs variables sous des contraintes générales. Elle utilise une méthode itérative. Chaque itération minimise une approximation du Lagrangien obtenu pour une linéarisation des contraintes. Une recherche linéaire est effectuée si la valeur initiale est loin de la solution.

b) Première méthode

Une première façon de résoudre le problème cité en début de paragraphe est de rechercher la solution du problème:

$$\begin{aligned} &\textbf{minimiser} \quad F(\delta) \\ &\textbf{sous} \quad\quad E(\delta) \leq ENEMIN + TOL \qquad (3.3) \\ &\qquad\quad 0 \leq \delta \leq 2\pi \qquad (*) \end{aligned}$$

où $E(\delta)$ est l'énergie calculée par la formule (3.1);
ENEMIN est le minimum d'énergie;
TOL est la tolérance;
$F(\delta)$ est la somme des distances au carré entre atomes à superposer des deux molécules.

Les fonctions $E(\delta)$ et $F(\delta)$ sont 2π-périodiques, nous restreignons donc la recherche du minimum sur une période (*).

Le problème (3.3) que nous avons à traiter est un problème de minimisation d'une fonction à une variable non linéaire, dérivable sous contraintes: une contrainte non linéaire à une variable et une contrainte de borne.

Nous utilisons pour le résoudre, la routine VF02AD de la librairie Harwell.

Inconvévient de la méthode

Nous avons amélioré le fitting flexible car l'angle optimal obtenu est tel que la conformation de la molécule après rotation a une énergie inférieure ou égale au SEUIL. Mais, rien ne dit que l'on pourra obtenir cette conformation car il se peut qu'il existe une barrière d'énergie entre l'angle nul et cet angle optimal.

Nous allons donc développer une deuxième méthode de résolution.

c) Deuxième méthode

Recherchons l'angle optimal pour lequel l'énergie est admissible de l'angle de départ à cet angle optimal: nous n'avons plus alors le problème d'une barrière d'énergie.

Nous supposons donc que l'énergie au départ est plus petite ou égale au seuil.

Nous résolvons le problème :

minimiser $F(\delta)$

sous $\delta_1 \leq \delta \leq \delta_2$ (3.4)

où δ_1 est la racine de la fonction $E(\delta)$ - ENEMIN - TOL la plus proche de zéro sur l'intervalle $[-\pi, 0]$

δ_2 est la racine de la fonction $E(\delta)$ - ENEMIN - TOL la plus proche de zéro sur l'intervalle $[0, \pi]$

La solution du problème (3.4) est l'angle optimal recherché.

En effet : 1) $[\delta_1, \delta_2] \subset [-\pi, \pi]$

d'où on recherche l'angle sur une période;

2) $E(0) \leq ENEMIN + TOL$

et par la définition de angles δ_1, δ_2, on a:

$\forall \Theta \in [\delta_1, \delta_2]$ $E(\Theta) \leq ENEMIN + TOL$ (*)

et δ_1, δ_2 sont les plus grandes valeurs telles que (*) soit vérifiée;

donc, l'énergie est admissible de l'angle nul à l'angle optimal.

Le calcul des racines δ_1, δ_2 se fera par la routine NB01AD de la librairie Harwell [8]. Cette routine cherche une racine réelle d'une fonction continue à une variable sur un intervalle [a,b]. La méthode utilisée se fait en deux étapes: elle recherche un intervalle $[\alpha,\beta]$ tel que la fonction en α est de signe opposé à la valeur de la fonction en β. L'algorithme recherche ensuite la racine par une méthode d'interpolation linéaire sur cet intervalle.

La solution du problème (3.4) se calcule analytiquement:

Si δ^*, solution de minimiser $F(\delta)$ est comprise entre δ_1 et δ_2,
alors δ^* est solution de (3.4).

Sinon la solution est δ_1 si $F(\delta_1) \leq F(\delta_2)$,
la solution est δ_2 si $F(\delta_2) \leq F(\delta_1)$.

En effet, la fonction ne possède qu'un minimum sur une période.
Si ce minimum appartient à l'intervalle $[\delta_1, \delta_2]$, il est la solution du problème.
Sinon, la solution est une des deux extrémités de l'intervalle.

3.3 Deuxième approche

Essayons de tenir compte de l'énergie d'une autre façon. Si nous portons l'énergie interne en fonction de l'angle de rotation, nous remarquons que celle-ci passe par une série de maxima et de minima. Toutefois, une molécule ne peut, à température ambiante, passer d'une conformation à l'autre que si la barrière énergétique ne dépasse pas un certain seuil.

Nous allons donc rechercher l'angle minimisant la somme des distances au carré pour lequel le passage de barrières, s'il y en a, pour aboutir à la conformation optimale est tel que chaque apport ne dépasse pas la quantité seuil. Connaissant l'angle minimisant la somme des distances au carré (voir le chapitre 2), nous allons rechercher un angle aussi proche que possible de celui-ci et pour lequel, partant de l'angle de rotation nul (ou de l'angle de torsion de départ), la conformation peut être obtenue selon notre critère.

3.3.1 Problème principal

Notations et conventions

Soit θ^* l'angle optimal calculé analytiquement. Par abus de langage, nous identifierons une conformation avec l'angle θ de rotation autour de la liaison simple qui engendre cette conformation. Ainsi parlerons-nous de la conformation optimale θ^*.

Soit $E(\theta)$ la fonction d'énergie. Nous savons que cette fonction est continue et 2π-périodique. Dès lors, nous pouvons restreindre notre recherche de l'angle optimal à l'intervalle $[0,2\pi]$. Ainsi, considérons les extréma de $E(\theta)$: θ_i, $i=1,...,m$ où $\forall i\ 0\leq\theta_i<2\pi$ et $\theta_i<\theta_{i+1}$. Soit $Z=\{0,\theta_1,...,\theta_m,2\pi\} = \{z_j\}_{j=0,n}$ où $n=m+1$. Sans perte de généralité, nous supposons que $\theta_1\neq 0$. Nous reviendrons plus tard sur le cas particulier où $\theta_1=0$.

Soit S kcal/mole ($S>0$) la quantité seuil : **étant donné deux conformations θ_1 et θ_2, nous pouvons passer de la conformation θ_1 à θ_2 si et seulement si $E(\theta_2)\leq E(\theta_1)+S$ c'est-à-dire si et seulement si la molécule n'a, à température ambiante, pas besoin de plus de S kcal/mole pour atteindre la conformation θ_2 à partir de θ_1.**

Stratégie

Soit z_0 la conformation initiale (c'est-à-dire celle correspondant à l'angle nul) et soit z_1 la conformation correspondant au premier extrémum non nul de l'énergie. Nous pouvons supposer, sans perte de généralité, que la conformation initiale n'est pas la conformation optimale (dans le cas contraire, le problème est résolu !). Supposons que l'angle optimum θ^* appartienne à l'intervalle $]z_0,z_1]$:

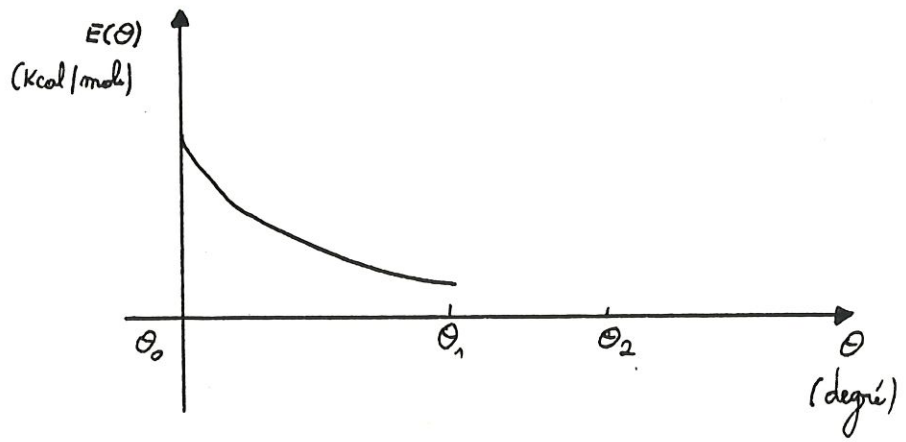
- si $E(\theta^*)\leq E(z_0)+S$, alors nous pouvons atteindre la conformation optimale et θ^*, la solution analytique, demeure l'angle optimal de ce problème sous contrainte d'énergie;

- si $E(\theta^*)>E(z_0)+S$, alors nous devons fournir plus de S kcal/mole pour atteindre la conformation optimale c'est-à-dire que le passage vers la conformation optimale est impossible. Dans ce cas, nous choisissons pour solution du fitting, l'angle $\underline{\theta}$ tel que $E(\underline{\theta})=E(z_0)+S$ ou encore, l'angle $\underline{\theta}$ qui résoud le problème $E(\theta)-E(\theta_0)-S=0$. Nous choisissons cet angle car $\underline{\theta}$ est l'angle admissible s'approchant le plus de l'angle optimal θ^*. En effet, rappelons que la fonction objectif $F(\theta)$ s'exprime comme une somme de fonctions cosinusoïdale et sinusoïdale de période 2π. Par conséquent, étant proche du minimum de $F(\theta)$ (c'est le cas puisque $\theta^* \in]z_0,z_1]$), plus on choisit un angle proche de θ^*, plus la valeur de la fonction objectif diminue.

Par contre, si l'angle optimum n'appartient pas à l'intervalle $]z_0,z_1]$, considérons l'intervalle $]z_1,z_2]$. Toutefois, pour pouvoir considérer cet intervalle, il est nécessaire qu'il soit accessible. En effet, nous devons pouvoir passer de la conformation initiale z_0 à la conformation z_1 correspondant à un extrémum de la fonction d'énergie $E(\theta)$:

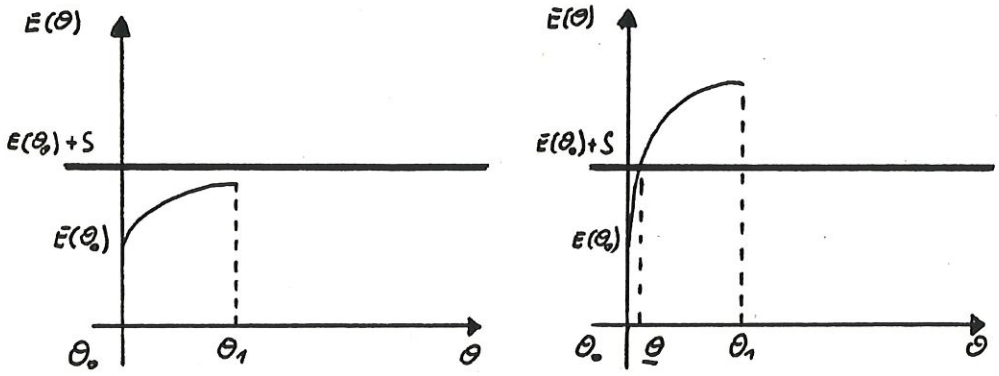
- <u>cas n°1</u> : si z_1 est un minimum, alors $E(z_1)<E(z_0)$. Dans ce cas, nous pouvons atteindre la conformation z_1 puisque nous ne devons fournir aucune énergie;

<u>Exemple</u>



- si z_1 est un maximum, alors $E(z_1)>E(z_0)$;
  . <u>cas n°2</u> : $E(z_1)\leq E(z_0)+S$; nous pouvons atteindre la conformation z_1 car l'énergie à fournir est inférieure ou égale à l'énergie maximale autorisée S;
  . <u>cas n°3</u> : $E(z_1)>E(z_0)+S$; nous ne pouvons pas atteindre la conformation z_1 car il faudrait fournir trop d'énergie. Dans ce cas, nous choisissons comme solution de ce problème sous contrainte d'énergie l'angle $\underline{\theta}$ racine de la fonction $E(\theta)-E(z_0)-S$.

<u>Exemples</u>



Dans les deux premiers cas, nous pouvons considérer l'intervalle $]z_1,z_2]$. Si l'angle optimum θ^* appartient à cet intervalle, nous devons encore vérifier comme précédemment que cet angle est accessible. Si c'est le cas, alors θ^* constitue la solution, sinon c'est l'angle $\underline{\theta}$ tel que $E(\underline{\theta})=E(z_1)+S$. Si θ^* n'appartient pas à $]z_1,z_2]$, nous devons prendre en considération l'intervalle $]z_2,z_3]$, à condition que cet intervalle soit accessible...

Nous continuons ce processus jusqu'à ce que nous atteignions l'angle optimum θ^* ou jusqu'à ce que nous nous heurtions à une barrière d'énergie supérieure à S. Dans ce dernier cas, la solution est l'angle $\underline{\theta}$ tel que $E(\underline{\theta})=E(z_i)+S$ où $\underline{\theta}\in]z_i,z_{i+1}]$.

<u>Remarques</u>

- Comme la fonction d'énergie $E(\theta)$ est 2π-périodique, $\exists j$ $0\leq j<n$ tel que $\theta^*\in]z_j,z_{j+1}]$. Ceci garantit que notre méthode s'effectue en un nombre fini de pas.

- Si $\theta_1=0$, alors nous définissons $Z=\{\theta_1,...,\theta_m,2\pi\}=\{z_j\}_{j=0,n}$ où n=m; la méthode reste valable.

La méthode que nous venons d'exposer revient à effectuer une rotation dans un sens bien précis : nous partons de l'angle nul et nous examinons les intervalles $]z_i,z_{i+1}]$ successifs avec $z_i<z_{i+1}$. Rien ne nous empêche d'effectuer cette méthode dans le sens inverse c'est-à-dire de partir de l'angle nul et de tendre vers l'angle -2π ou plutôt, comme la fonction $E(\theta)$ est périodique, de partir de l'angle 2π et de tendre vers 0. Il se peut que nous atteignions l'angle θ^* ou un autre angle $\hat{\theta}$. Remarquons que $\underline{\theta}<\theta^*$ et $\theta^*<\hat{\theta}$ et donc $\underline{\theta}\neq\hat{\theta}$. Il ne reste plus qu'à comparer $F(\underline{\theta})$ et $F(\hat{\theta})$ pour savoir lequel des deux angles "minimise" la fonction objectif.

Remarque

Si $\hat{\theta}$ constitue la solution de ce problème, la véritable solution est en fait $\hat{\theta}-2\pi$ puisque $\hat{\theta}$ n'est pas accessible à partir de la configuration initiale c'est-à-dire de l'angle nul. En effet, $\hat{\theta}$ est obtenu en partant de 2π et non de 0 !

En conclusion, nous allons appliquer deux fois la méthode : une fois en partant de 0 et en faisant croître l'angle, une fois en partant de 2π et en faisant décroître l'angle. Il suffit alors de choisir pour solution du fitting sous contrainte d'énergie la meilleure solution :

soit S_1 la solution obtenue en partant de 0, S_2 la solution obtenue en partant de 2π et S^* la solution du fitting sans contrainte d'énergie :

- si $S_1=S_2$, alors $S^*=S_1$;
- sinon si $F(S_1)\leq F(S_2)$, alors $S^*=S_1$;
- sinon $S^*=S_2-2\pi$.

3.3.2 Sous-problèmes

Pour pouvoir appliquer notre méthode, nous devons connaître les extréma θ_i de la fonction d'énergie. De plus, nous devons être capables de rechercher la racine $\underline{\theta}$ de la fonction $E(\theta)-E(z_i)-S$ sur un intervalle $]z_i,z_{i+1}]$. Commençons par résoudre ce problème.

3.3.2.1 Racine d'une fonction continue

De manière générale, recherchons la racine d'une fonction continue appartenant à l'intervalle]a,b[. Pour cela, utilisons la méthode du balayage [9].

Cette méthode attribuée à Horner est fondée sur la propriété de Darboux :

si f est une fonction continue sur un intervalle compact [a,b] et à valeurs réelles, et si f(a) et f(b) sont de signes opposés, f s'annule au moins une fois sur]a,b[.

Cette propriété est un corollaire du théorème de la valeur intermédiaire [10] :

une fonction f continue sur un intervalle compact [a,b] à valeurs réelles prend au moins une fois toute valeur comprise entre f(a) et f(b).

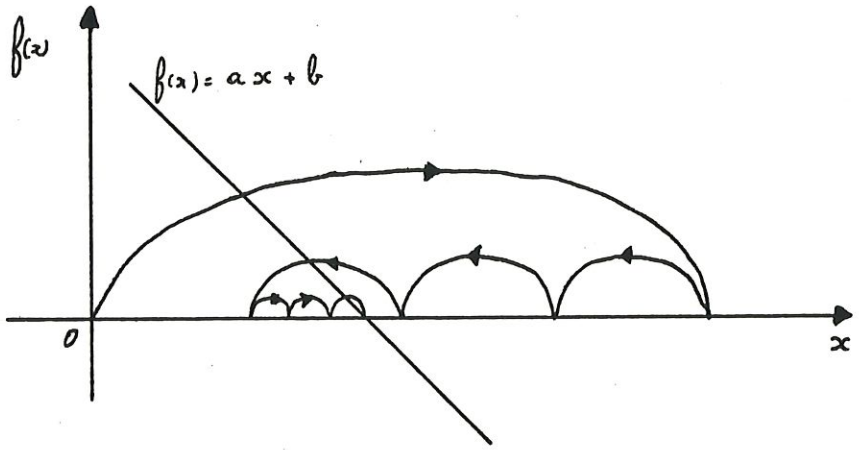
L'algorithme présenté ici assure alternativement un balayage "de gauche à droite" et "de droite à gauche", à des pas, h, de plus en plus fins. Le pas h est l'espace entre "les dents du râteau" : si les dents sont trop espacées, on risque de manquer certaines racines; si les dents sont trop resserrées, le ratissage dure longtemps.

Description de la méthode

A partir de a et du pas h ($h<|b-a|$), on calcule le produit $f(a).f(a+h)$. Si ce produit est nul, il y a arrêt car, a n'étant pas racine, a+h l'est. Si le produit est positif, on continue le balayage. S'il est négatif, on commence un nouveau balayage à partir de a+h dans l'autre sens, au pas -h/p. Si p est 2, la méthode du balayage est encore appelée dichotomie. Le calcul est arrêté si une racine a été trouvée, ou si le

nombre n de "coups de balai" initialement imposé a été réalisé. L'erreur absolue sur la racine est alors au plus égale à h/p^{n-1}.

Illustration :



Montrons que nous pouvons utiliser la méthode du balayage pour résoudre notre problème : vérifions les hypothèses de la propriété de Darboux.

Preuve

1. Nous savons que la fonction $E(\theta)-E(z_i)-S$ est continue. De plus, l'intervalle $[z_i,z_{i+1}]$ est compact et $E(\theta)-E(z_i)-S$ est à valeurs réelles.

2. Nous devons rechercher la racine de $E(\theta)-E(z_i)-S$ si, lorsque nous examinons l'intervalle $]z_i,z_{i+1}]$, nous rencontrons l'un des deux cas suivants :
- l'angle optimal θ^* appartient à cet intervalle et $E(\theta^*)>E(z_i)+S$;
- θ^* n'appartient pas à cet intervalle et $E(z_{i+1})>E(z_i)+S$.

Dès lors, $E(z_i)-E(z_i)-S=-S < 0$.
De plus, dans le premier cas, $E(z_{i+1})\geq E(\theta^*)>E(z_i)+S$ (z_{i+1} maximum local).
D'où, dans les deux cas, $E(z_{i+1})-E(z_i)-S>0$.
La fonction $E(\theta)-E(z_i)-S$ prend donc bien des valeurs non nulles et de signes opposés en z_i et z_{i+1}.

Remarque

Par définition de la méthode, nous sommes assurés de l'unicité de la racine.

3.3.2.2 Recherche et identification des extréma

Par définition, nous savons que θ_i est un extrémum local si et seulement si $E'(\theta)_{|\theta i}=0$ et $E''(\theta)_{|\theta i}\neq 0$. Nous allons donc rechercher les racines de la fonction dérivée première. Ensuite, nous éliminerons les éventuels points d'inflexion (c'est-à-dire les θi tels que $E''(\theta)_{|\theta i}=0$). Enfin, nous identifierons les extréma.

Stratégie

Comme la fonction $E(\theta)$ est 2π-périodique, nous pouvons limiter notre recherche à l'intervalle $[0,2\pi]$, ou pour plus de facilité, à l'intervalle [0,360]. Des conversions en radian seront alors nécessaires. Nous avons choisi l'intervalle [0,360] car nous pouvons plus facilement le subdiviser. En effet, étant donné la forme de la fonction d'énergie et par expérience, nous pouvons faire l'hypothèse que *deux extréma sont "distants" de plus de dix degrés*.

Soit $G : [0,360] \rightarrow \mathbb{R}$ la fonction dérivée première d'énergie.

$\theta \rightarrow E'(\theta)$

Par souci de simplification, nous supposons que l'angle θ a été transformé en radian et est encore noté θ.

Soit $Z=\{0,\theta_1,\ldots,\theta_m,2\pi\}=\{z_j\}_{j=0,n}$ où $n=m+1$ et θ_i est extrémum de $E(\theta)$. Si $\theta_1=0$, alors $Z=\{\theta_1,\ldots,\theta_m,2\pi\}$. En fait, c'est cet ensemble Z que nous devons construire pour pouvoir utiliser la méthode exposée précédemment.

Construisons cet ensemble Z :

Considérons les trente-six intervalles]0,10],]10,20], ..., et]350,360], c'est-à-dire les intervalles de la forme $]\delta_i,\delta_{i+1}]$ où $\delta_{i+1}=\delta_i+10$ et $\delta_1=0$.

+ Initialisation : $j=1$, $z_j=0$

+ Itération :
  Examinons l'intervalle $]\delta_i,\delta_{i+1}]$ avec $G(\delta_i)\neq 0$ et $i\leq 36$:
  - si $G(\delta_{i+1})=0$, alors δ_{i+1} est une racine. Par conséquent, $j=j+1$ et $z_j=\delta_{i+1}$. Nous examinons l'intervalle suivant. Celui-ci n'est pas l'intervalle $]\delta_{i+1},\delta_{i+2}]$ car, puisque deux extréma sont séparés par plus de dix degrés, le prochain extréma n'appartient pas à l'intervalle $]\delta_{i+1},\delta_{i+2}]$. Nous examinons donc l'intervalle $]\delta_{i+2},\delta_{i+3}]$ (si $i+2\leq 36$). Remarquons que $G(\delta_{i+2})\neq 0$ puisque $\delta_{i+2}=\delta_{i+1}+10$ et $G(\delta_{i+1})=0$;
  - si $G(\delta_{i+1})\neq 0$, alors
    . ou $G(\delta_i).G(\delta_{i+1})<0$; nous avons trouvé un intervalle vérifiant les hypothèses d'application de la méthode du balayage : $G(\delta_i)\neq 0$, $G(\delta_{i+1})\neq 0$, $G(\delta_i)$ et $G(\delta_{i+1})$ sont de signes opposés. Le balayage nous donne la racine z_j avec $j=j+1$. Grâce à notre hypothèse, nous savons que cette racine est unique. Nous passons ensuite à l'intervalle suivant si $i+1\leq 36$.
    . ou $G(\delta_i).G(\delta_{i+1})>0$; alors l'intervalle $]\delta_i,\delta_{i+1}]$ ne possède pas de racine. Nous passons à l'intervalle suivant si $i+1\leq 36$.

+ Comme le dernier élément de Z doit être 360, si $z_j\neq 360$, alors $j=j+1$ et $z_j=360$.

Il ne reste plus qu'à éliminer de Z les points d'inflexions et d'identifier les extréma :

- si $E''(\theta)_{|\theta_i}=0$, alors θ_i est un point d'inflexion. Dans ce cas, nous éliminons cet angle de Z;
- si $E''(\theta)_{|\theta_i}<0$, alors θ_i est un maximum;
- si $E''(\theta)_{|\theta_i}>0$, alors θ_i est un minimum.

Chapitre IV

Fitting flexible sur plusieurs axes de rotation

Ce chapitre traite d'une méthode qui utilise plusieurs liaisons simples comme axes de rotation.

Introduction

Le fait d'avoir plusieurs axes de rotation permet une plus grande modification de la conformation de la molécule mobile. La superposition de certains atomes de la molécule mobile sur des atomes équivalents de la molécule de référence en sera donc facilitée.

4.1 Notations

Nous rappelons:

$X=\{x_i,\ i=1,...,R\}$ la molécule de référence constituée de **R** atomes x_i;

$Y=\{y_j,\ j=1,...,B\}$ la molécule mobile constituée de **B** atomes y_j;

$(x_{i1},x_{i2},x_{i3})^t$ les coordonnées de l'atome x_i de la molécule de référence;

$(y_{j1},y_{j2},y_{j3})^t$ les coordonnées de l'atome y_j de la molécule mobile.

$AFE=\{x_i \in X,\ i=1,...,E\}$ l'ensemble des atomes à superposer de la molécule de référence;

$AME=\{y_i \in Y,\ i=1,...,E\}$ l'ensemble des atomes à superposer de la molécule mobile.

Ces deux ensembles ne sont pas vides.
Nous superposons le i° élément de **AME** avec le i° élément de **AFE**, i=1,...,E.

Nous considérons **n** axes de rotation.

4.2 Problème

Nous recherchons les angles de rotation autour des liaisons simples minimisant la somme des distances au carré entre les atomes à superposer.

Le problème s'écrit:

$$\text{minimiser } F(\delta_1,\ldots,\delta_n) = \Sigma_{i=1,E} ([x_{i1} - y'_{i1}]^2 + [x_{i2} - y'_{i2}]^2 + [x_{i3} - y'_{i3}]^2) \qquad (4.1)$$

où y'_i est l'atome y_i obtenu après rotation de chacune des parties mobiles autour de leur axe.

Les seules inconnues étant les angles de rotation $\delta_1,\ldots, \delta_n$, la fonction objectif dépend de n variables.

Nous devons donc résoudre un problème de minimisation d'une fonction non linéaire à n variables.

4.3 Résolution

Pour un axe de rotation, en déplaçant la molécule mobile pour faire passer l'axe par l'origine, nous avions l'expression: $y'_i = A\, y_i$ où A est une matrice de rotation.

Pour ce problème, nous n'avons plus une relation exprimant y'_i en fonction de y_i; néanmoins, connaissant la valeur des angles, on peut le calculer.

Plusieurs approches ont été envisagées.

1) Nous aurions pu résoudre ce problème en utilisant une routine de minimisation d'une fonction non linéaire à plusieurs variables, mais nous voulions une résolution plus spécifique.

2) C'est pourquoi nous avons orienté nos recherches dans le domaine de la robotique.

En effet, on peut définir un robot comme un ensemble formé d'une structure mécanique supportant un organe terminal et des actionneurs servant à agir sur la structure pour en modifier la configuration et donc la situation de l'organe terminal. Ce robot a pour but de déplacer l'organe terminal en un endroit précis, il doit donc superposer deux points, l'un étant fixe, l'autre pouvant être modifié par des rotations ou des translations.

Malgré les ressemblances, cette approche a été abandonnée car elle ne s'adaptait pas bien à notre problème.

3) Le problème (4.1) est un problème de moindres carrés non linéaires.

Nous avons utilisé pour le résoudre la routine LMDIF1 de la librairie Minpack [9]. Cette routine minimise la somme des carrés de M fonctions à N variables par une méthode qui est une modification de l'algorithme de Levenberg-Marquart.

Dans le but d'utiliser cette routine, réécrivons le problème:

$$\textbf{minimiser } F(\delta_1,\ldots,\delta_n) = \sum_{i=1,E} F_i$$

$$\text{où } F_i = [\, x_{i1} - y'_{i1} \,]^2 + [\, x_{i2} - y'_{i2} \,]^2 + [\, x_{i3} - y'_{i3} \,]^2$$

y'_i est évalué à partir de y_i en effectuant chaque rotation les unes après les autres et en ne l'effectuant que si l'atome i est dans la partie mobile par rapport à la liaison simple autour de laquelle la rotation s'effectue.

Pour utiliser la routine, nous supposerons que le nombre de fonctions F_i, c'est-à-dire le nombre d'atomes à superposer, est supérieur ou égal au nombre de variables, c'est-à-dire au nombre d'axes de rotation.

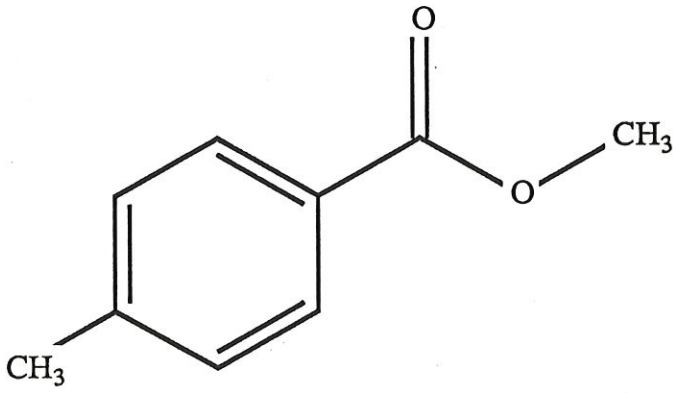
Chapitre V

Vérifications

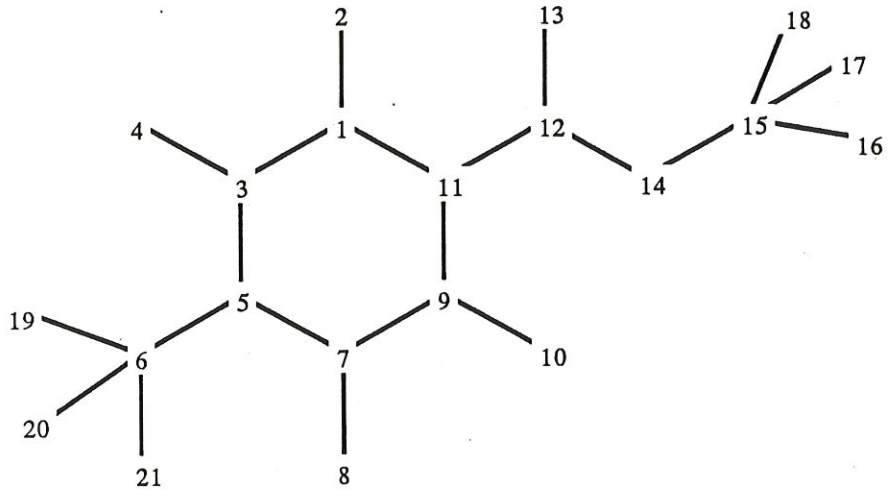
Dans ce chapitre, nous allons exposer une méthode adaptée par le laboratoire de Chimie moléculaire structurale et Radiocristallographie (C.M.S.) des Facultes Universitaires N-D de la Paix de Namur qui permet de déterminer la partie mobile et la partie fixe d'une molécule par rapport à un axe de rotation donné. De plus, nous allons procéder à certaines vérifications nécessaires pour que les méthodes de comparaison moléculaire exposées precédemment puissent fonctionner. Ces vérifications portent d'une part sur la connexité de la molécule et d'autre part sur l'admissibilité de la liaison simple autour de laquelle nous effectuons une rotation.

Introduction

Pour resoudre les problemes de connexite de la molécule et d'admissibilité de la liaison simple autour de laquelle nous effectuons une rotation, nous faisons appel à la theorie des graphes. En effet, une molécule peut être vue de manières fort différentes selon les aspects et les propriétés que nous désirons mettre en évidence. Si nous nous intéressons à la molécule comme ensemble d'atomes liés entre eux, des schémas tels que le suivant nous sont très utiles :



Ou encore , de manière plus simplifiee :



Remarque

Rappelons qu'une liaison double ou triple est en realite une seule liaison chimiquement plus forte.

Un tel schema n'est pas representatif de la forme tridimensionnelle de la molécule et réduit celle-ci à la seule information concernant sa connexité :

- les atomes sont representés par des points numérotés en faisant abstraction de leur nature chimique;
- toutes les liaisons sont représentées par des traits sans tenir compte de leur nature.

Nous avons donc abandonné les caractéristiques topographiques et topologiques de la molécule pour nous intéresser uniquement à sa connexité. Nous pouvons donc concevoir une molécule comme un multigraphe simple, simplement connexe.

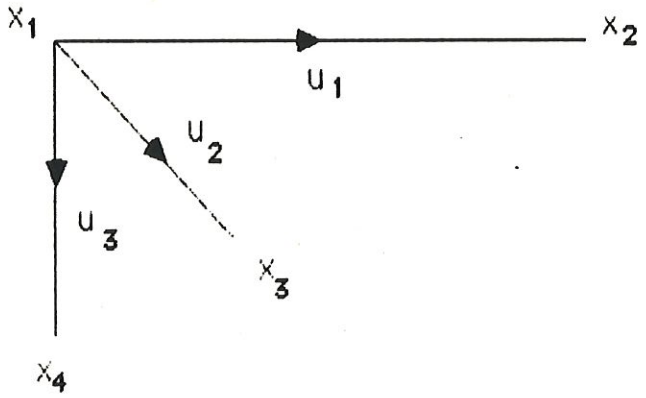
Avant de poursuivre, il serait bon de préciser le vocabulaire que nous utiliserons dans ce chapitre [11].

5.1. Concepts

Un **graphe** est un couple G=(X,U) constitué par
- un ensemble fini $X=\{x_1,x_2,\ldots,x_n\}$ dont les éléments sont appelés **sommets** du graphe,
- une famille $U=(u_1,u_2,\ldots,u_n)$ d'éléments du produit cartésien $X\times X=\{(x,y),x\in X,y\in X\}$ appelés **arcs**.

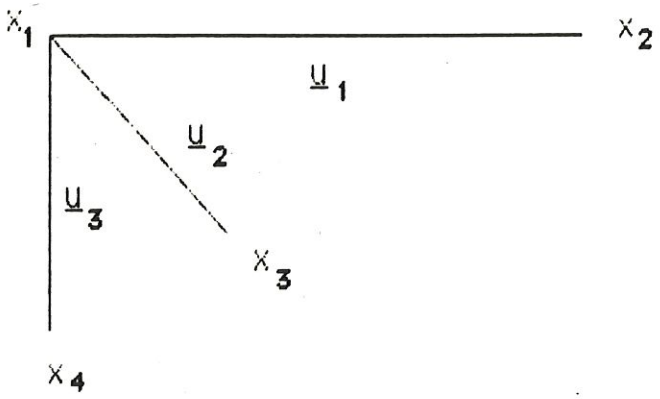
Un graphe est représenté par un schema dans $\mathbb{R}^2$ où les sommets sont représentés par des points et les arcs sont représentés par des "branches" orientées reliant deux sommets.

Exemple



Un **multigraphe** est un couple $\underline{G}=(X,\underline{U})$ constitué par
- un ensemble fini $X=\{x_1,x_2,\ldots,x_n\}$ dont les éléments sont appelés **sommets** du multigraphe,
- une famille $\underline{U}=(\underline{u}_1,\underline{u}_2,\ldots,\underline{u}_n)$ d'elements appelés **arêtes** où chaque arête $\underline{u}_i=(\underline{x}_i,\underline{x}_j)$ est un ensemble non ordonné formé de deux éléments (i.e. une paire) $x_i\in X$ et $x_j\in X$ et représentés par une "branche" non orientée reliant x_i et x_j.

Exemple



Remarque

Une molécule est un ensemble d'atomes que nous pouvons considérer comme les sommets d'un multigraphe dont les arêtes sont les liaisons chimiques. Il s'agit bien d'un multigraphe puisque les liaisons chimiques ne sont pas orientées.

On appelle **ordre** du (multi)graphe le nombre n de sommets.

Un (multi)graphe est un **p-(multi)graphe** lorsque $\forall$ i,j$\in$n, le nombre maximal d'(arêtes) arcs (x_i,x_j) est p.

On appelle **boucle** un (arête) arc de la forme (x_i,x_i).

Un (multi)graphe est **simple** si et seulement si c'est un 1-(multi) graphe sans boucle.

Remarque

Une molécule est un multigraphe simple : toute paire d'atomes est au plus liée par une liaison et aucun atome n'est lié à lui-même.

Soit G=(X,U) un graphe.
Si $u=(x_i,x_j)$ est un arc de G, on dit que x_i et x_j sont respectivement les **extrémités initiale** et **terminale** de u.

Deux sommets sont **adjacents** si il existe un arc les reliant.

Deux arcs sont **adjacents** si ils sont distincts et admettent une extrémité commune.

Un **chemin** μ est une suite d'arcs $\mu=(u_1,u_2,...,u_i,u_{i+1},...,u_q)$ telle que l'extrémité terminale de u_i coïncide avec l'extrémité initiale de u_{i+1}. La longueur du chemin μ est le nombre **q** d'arcs qui composent le chemin.

Un circuit est un chemin $\sigma=(u_1,u_2,...,u_q)=(x_1,x_2,...,x_1)$ tel que l'extremite terminale du dernier arc u_q est l'extrémité initiale du premier arc u_1.

On appelle :

ascendant de $x_j \in X$ tout sommet $x_i \in X$ duquel part un chemin aboutissant en x_j;

précédent de $x_j \in X$ tout sommet $x_i \in X$ duquel part un chemin de longueur 1 aboutissant en x_j;

descendant de $x_i \in X$ tout sommet $x_j \in X$ auquel aboutit un chemin partant de x_i;

suivant de $x_i \in X$ tout sommet $x_j \in X$ auquel aboutit un chemin de longueur 1 partant de x_i;

racine tout sommet $x_i \in X$ qui est ascendant de tous les autres sommets du graphe.

On appelle **graphe partiel** de G=(X,U) engendré par V où $V \subset U$, le graphe (X,V); on a donc éliminé de G les arcs de U\V.

Soit $\underline{G}=(X,\underline{U})$ un multigraphe.
Si $\underline{u}=(\underline{x_i},\underline{x_j}) \in \underline{U}$, on dit que x_i et x_j sont les **extrémités** de l'arête $\underline{u}$ et que x_i et x_j sont **adjacents**.
Deux arêtes sont **adjacentes** si elles ont une extrémité en commun.

Une arête est **incidente** à un sommet x_i si ce n' est pas une boucle et si ce sommet est l'une de ses extrémités.

Le **degré** d'un sommet est le nombre d'arêtes qui lui sont incidentes.

Remarque

Bien qu'il existe des molécules dont certains atomes sont liés a plus de quatre autres atomes, nous ne considérons que celles dont le degré des atomes est au plus quatre. C'est le cas pour une majorité de composés organiques constitués essentiellement d'atomes de carbone, d'azote, d'oxygène et d'hydrogène.

Une **chaîne** est une suite $\underline{\mu}=(\underline{u}_1,\ldots,\underline{u}_{i-1},\underline{u}_i,\underline{u}_{i+1},\ldots,\underline{u}_q)$ d'arêtes telle que l'arête $\underline{u}_i$ ($1<i<q$) est reliée à $\underline{u}_{i-1}$ par une de ses extrémités et à $\underline{u}_{i+1}$ par l'autre. De plus, si il y a répétition des arêtes, chaque arête repetée est parcourue dans le même sens.

Un **cycle** $\underline{\sigma}$ est une chaîne qui part d'un sommet x_i et aboutit au même sommet x_i.

Un cycle est **simple** lorsqu'il ne passe pas plus d'une fois par la même arête.

Un cycle est **élémentaire** lorsqu'il ne passe pas plus d'une fois par chacun de ses sommets (sauf le premier et le dernier qui coïncident).

On appelle **multigraphe partiel** de $\underline{G}=(X,\underline{U})$ engendré par $\underline{V} \subset \underline{U}$, le multigraphe $(X,\underline{V})$; on a donc éliminé de $\underline{G}$ les arêtes de $\underline{U} \setminus \underline{V}$.

Un (multi)graphe est **simplement connexe** si et seulement si il existe une chaîne reliant toute paire de sommets.

Remarque

Une molécule est donc un multigraphe simple, simplement connexe : tous les atomes sont liés entre eux.

Soit G=(X,U) un graphe.
G est un **arbre** si et seulement si il est simplement connexe et sans cycle.
G est une **arborescence** si et seulement si il est un arbre et admet une racine.
H est une **arborescence maximale** si et seulement si H est un graphe partiel de G et une arborescence.

Les arcs appartenant à ce graphe partiel H sont appeles **branches**; les arcs de G\H sont appelés **cordes** .

Soit G=(X,U) un graphe.
Pour chaque sommet $x_i \in X$, nous pouvons établir une liste contenant tous les sommets x_j adjacents au sommet x_i. Une telle liste est appelée **liste d'adjacence** du sommet x_i. L'ensemble de ces listes pour chaque sommet de G est appelé **structure d'adjacence** de G.

Remarque

Si G est un multigraphe, chaque arête (x_i,x_j) est représentée deux fois dans la structure d'adjacence : une fois pour x_i et une fois pour x_j. Si G est un graphe, chaque arc est représenté une seule fois : le sommet x_j apparaît dans la liste d'adjacence du sommet x_i. Un graphe simple peut avoir beaucoup de structures d'adjacence. En fait, chaque numérotation des sommets donne une structure d'adjacence unique et chaque structure d'adjacence correspond à une numérotation unique de chaque sommet.

On appelle **matrice de distance** d'un (multi)graphe une matrice D telle que D(I,J) représente le nombre d'(arêtes) arcs du plus court chemin entre les sommets I et J.

Remarque

La matrice de distance d'un multigraphe est symétrique.

5.2 Détermination de la partie fixe et de la partie mobile d'une molécule

5.2.1 Méthode

Nous avons vu qu'un inconvénient du fitting développé dans le chapitre 2 est que l'utilisateur doit déterminer lui-même quelle est la partie qui reste fixe lors de la rotation de la molécule autour d'une liaison simple et quelle est la partie mobile. L'utilisateur se voit donc dans l'obligation de communiquer à l'ordinateur de nombreuses données relativement sophistiquées sans moyen aisé de vérification. Le laboratoire de Chimie moléculaire structurale et Radiocristallographie (C.M.S.) des Facultés Universitaires N-D de la Paix de Namur a développé une méthode pratique palliant à cet inconvénient [12],[13]. Nous nous proposons d'exposer brièvement cette technique.

Nous savons qu'une molécule peut être considérée comme un multigraphe. Il y a donc un sens à parler de la matrice de distance D d'une molécule. Rappelons que pour une molécule, D(I,J) représente le nombre de liaisons formant le plus court chemin entre l'atome I et l'atome J.

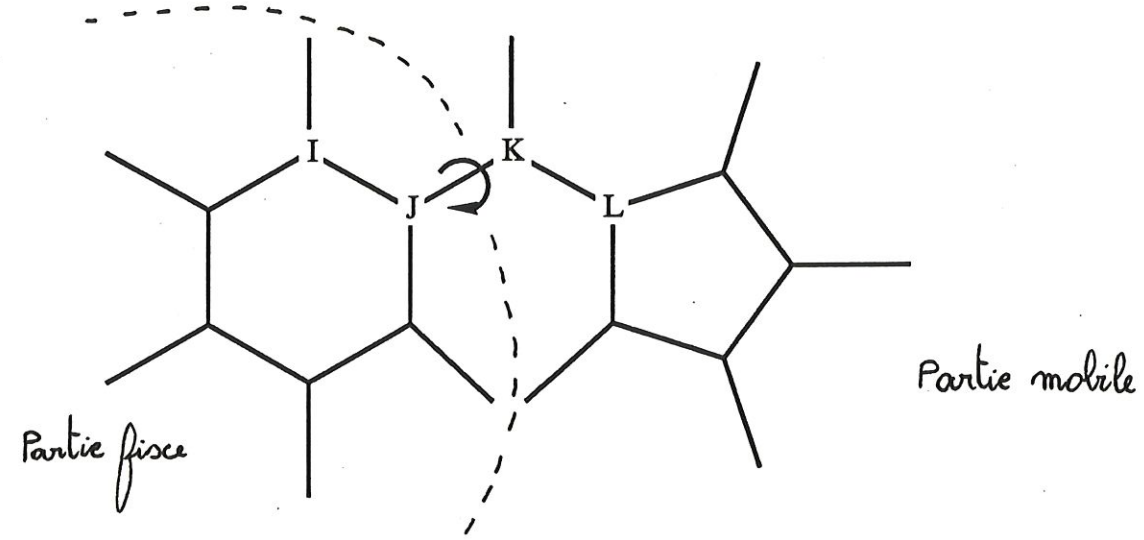
Supposons que nous désirons effectuer une rotation autour de la liaison simple déterminée par les atomes J et K. Nous définissons l'angle de torsion correspondant par les atomes I, J, K et L où I est le premier atome adjacent à J et situé dans la partie de la molécule qui reste fixe durant la rotation, et L est le premier atome adjacent à K situé dans la partie mobile. Dès lors, tout atome X appartenant à la partie mobile doit satisfaire la relation suivante :

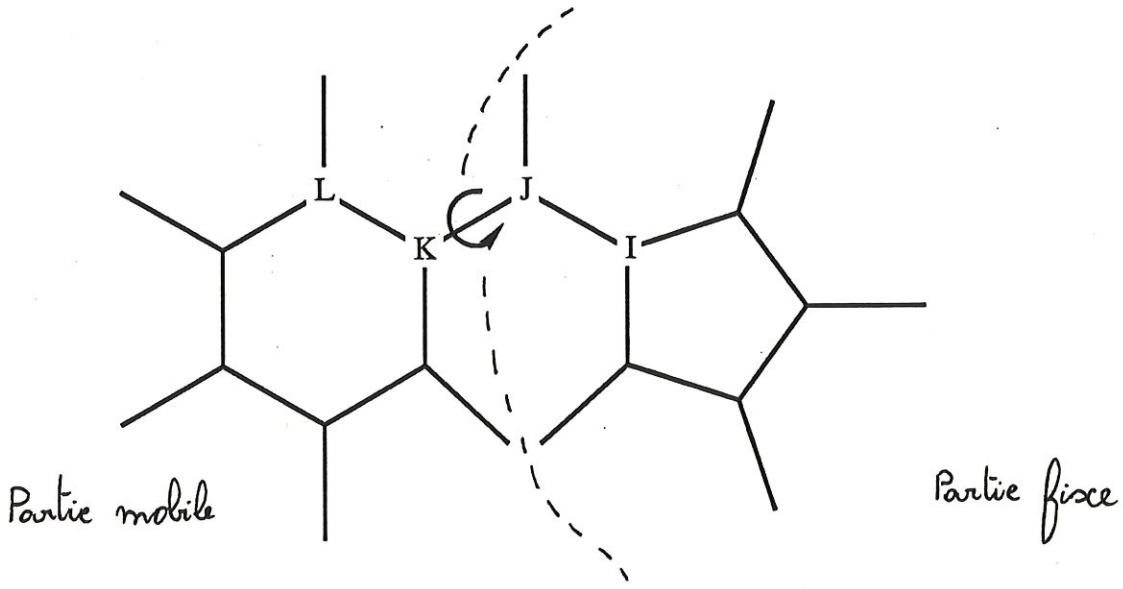
$$D(J,X)-D(K,X)=+1$$

Les atomes de la partie fixe satisfont la relation :

$$D(J,X)-D(K,X)=-1$$

Exemples


I
J
K
L
Partie fixe
Partie mobile


L
K
J
I
Partie mobile
Partie fixe

5.2.2 Algorithme rapide pour le calcul de la matrice de distance d'une molécule

La matrice de distance d'un graphe est généralement engendrée en utilisant les puissances de la matrice d'adjacence. La matrice d'adjacence est une matrice A telle que A(I,J) est unitaire si le sommet I est adjacent au sommet J ; sinon A(I,J) vaut 0. Puisque la distance séparant deux sommets adjacents est 1, pour toutes les paires de sommets pour lesquelles A(I,J)=1, D(I,J) doit valoir également 1. Evidemment, D(I,I) vaut 0.

L'élément (I,J) de A^2 vaut $\sum_{K=1,n} A(I,K).A(K,J)$ où n est l'ordre du multi-graphe. Dès lors, nous voyons que l'élément (I,J) de A^2 représente le nombre de chemins de longueur 2 partant du sommet I et aboutissant au sommet J. C'est pourquoi, si l'élément (I,J) de A^2 n'est pas nul, il y a (au moins) un chemin de longueur 2 entre ces deux sommets et donc D(I,J)=2 (si A(I,J)=0). De même, si A(I,J) =0 et A^2(I,J)=0, alors si A^3(I,J)≠0, D(I,J)=3. De cette manière, en prenant les puissances successives de la matrice d'adjacence, nous pouvons construire progressivement la matrice de distance.

Toutefois, M. Bershon a développé un algorithme plus rapide [14]. C'est cet algorithme que nous utilisons pour créer la matrice de distance de la molécule.

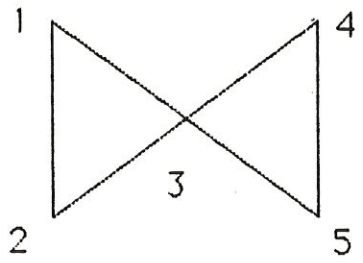
Formons une liste "liée" d'éléments : chaque élément contient un pointeur vers l'élément suivant et un pointeur vers l'élément précédent. En outre, chaque élément contient deux sommets adjacents I et J. Nous construisons cette liste enchaînée en passant en revue la structure d'adjacence de la molécule. Lorsque nous avons terminé la liste, nous lions le dernier élément au premier, formant ainsi une liste circulaire. Pour chaque élément ainsi créé, nous assignons la valeur 1 à D(I,J) (et D(J,I)) puisque I et J sont adjacents.

Pour chaque élément de la liste circulaire, nous examinons les sommets adjacents au sommet I (ces sommets appartiennent, par définition, à la liste d'adjacence de I). Supposons qu'un des sommets adjacents soit le sommet K. Si D(K,J) est inconnu, nous pouvons conclure que D(K,J) est égal à D(I,J)+1. Nous construisons alors un nouvel élément (K,J) et nous l'insérons avant l'élément (I,J). En d'autres termes, le pointeur vers l'arrière du nouvel élément pointe vers l'élément qui précédait l'élément (I,J) et le pointeur vers l'avant pointe vers l'élément (I,J).

Ensuite, nous examinons les sommets adjacents au sommet J. Supposons qu'un des sommets adjacents à J soit le sommet L. Si D(L,I) est inconnu, nous concluons que D(L,I)=D(J,I)+1. Nous ajoutons alors l'élément (L,I) dans la liste circulaire, juste derrière l'élément (I,J). Lorsque nous avons fait cela pour tous les sommets adjacents à I et pour tous les sommets adjacents à J, alors nous enlevons l'élément (I,J) de la liste circulaire et nous passons à l'élément suivant. Lorsque nous avons tourné autour du cercle un certain nombre de fois, ajoutant les paires d'atomes séparés par X+1 liaisons et enlevant les éléments représentant des paires d'atomes séparés par X liaisons, nous avons enlevé le dernier élément de la liste. Ce dernier élément est la paire d'atomes séparés par la distance maximum.

Exemple

Multigraphe



Structure d'adjacence

1	2	3		
2	1	3		
3	1	2	4	5
4	3	5		
5	3	4		

Chemins de longueur minimale 1

-(1,2)-(1,3)-(2,3)-,(3,4)-(3,5)-(4,5)-

0 1 1 0 0
1 0 1 0 0
1 1 0 1 1
0 0 1 0 1
0 0 1 1 0

Chemins de longueur minimale 2

-(1,4)-(1,5)-(2,4)-(2,5)-

0 1 1 2 2
1 0 1 2 2
1 1 0 1 1
2 2 1 0 1
2 2 1 1 0

5.3 Vérifications de la connexité de la molécule

Nous avons vu précédemment qu'une molécule peut être considérée comme un multigraphe G simple, simplement connexe et dont le degré de ses sommets est au plus quatre. Dans ce paragraphe, nous nous proposons de vérifier si les molécules données par l'utilisateur possèdent ces trois propriétés.

5.3.1 Le degré de chaque sommet est au plus quatre

Nous allons vérifier cette propriété en construisant la structure d'adjacence du multigraphe G. Pour construire cette structure, nous avons besoin de trois renseignements fournis par l'utilisateur, à savoir:
- l'ordre du multigraphe *NSO* c'est-à-dire le nombre d'atomes de la molécule,
- le nombre d'arêtes *NAR* du multigraphe c'est-à-dire le nombre de liaisons de la molécule.
- l'ensemble *BOND* des paires de sommets définissant les liaisons; BOND(1,I) et BOND(2,I) désignent les deux sommets définissant la $I^{ème}$ arête du multigraphe.

Algorithmique

Désignons par *ADJL* la matrice représentant la structure d'adjacence du multigraphe : la $I^{ème}$ colonne de cette matrice est la liste d'adjacence du $I^{ème}$ sommet ($1 \leq I \leq NSO$). Si le degré du sommet I est inférieur à quatre, le(s) dernier(s) élément(s) de la liste d'adjacence sera(seront) nul(s). Nous pouvons utiliser cette convention puisqu'aucun sommet ne porte le numéro 0.

Stratégie

Nous passons en revue chaque arête c'est-à-dire chaque paire (BOND(1,I),BOND(2,I)), I=1,NAR. Pour chaque paire, nous complétons la liste d'adjacence de ces sommets si c'est possible. En effet, rappelons que nous devons vérifier le degré de chaque sommet : le degré doit être inférieur ou égal à quatre. Dès lors, si nous devons compléter la liste d'adjacence d'un sommet dont le degré est déjà quatre, ce sommet aura un degré strictement supérieur à quatre. Dans ce cas, nous devons arrêter la construction de la structure d'adjacence et signaler à

l'utilisateur qu'il a introduit une mauvaise molécule.

De manière plus approfondie : pour chaque paire, tant que le degré de chaque sommet est correct,

- nous complétons la liste d'adjacence du sommet BOND(1,I) :
  - . si le degré de ce sommet est inférieur à quatre, alors nous pouvons compléter sa liste d'adjacence :
    - . nous augmentons d'une unité le degré de ce sommet,
    - . nous ajoutons le sommet BOND(2,I) dans la liste d'adjacence du sommet BOND(1,I);
  - . sinon nous arrêtons.
- nous complétons la liste d'adjacence du sommet BOND(2,I) :
  - . si le degré de ce sommet est inférieur à quatre, alors nous pouvons compléter sa liste d'adjacence :
    - . nous augmentons d'une unité le degré de ce sommet,
    - . nous ajoutons le sommet BOND(1,I) dans la liste d'adjacence du sommet BOND(2,I);
  - . sinon nous arrêtons.

5.3.2 Le multigraphe est simplement connexe

Nous devons vérifier que le multigraphe est simplement connexe c'est-à-dire qu'il existe une chaîne reliant toute paire de sommets.

Pour effectuer cette vérification, nous nous basons sur un algorithme développé par R. Tarjan [15]. Cet algorithme engendre à partir du multigraphe G, un graphe P jouissant d'une structure particulière. C'est cette structure particulière qui va nous permettre de tester la connexité du multigraphe. Tarjan a écrit son algorithme intitulé **recherche en profondeur** (depth-first search) principalement pour améliorer la recherche des composantes fortement connexes d'un graphe et pour rechercher les composantes biconnexes d'un multigraphe.

En fait, nous avons été initialement amenés à considérer cet algorithme pour résoudre un autre problème à savoir comment tester si une arête appartient, ou non, à un cycle. Ce problème sera traité au paragraphe 5.4.3.

5.3.2.1 Recherche en profondeur

La méthode de recherche en profondeur est une technique qui a été fréquemment utilisée pour rechercher la solution de problèmes en théorie combinatoire et en intelligence artificielle mais dont les propriétés n'ont pas été analysées complètement [16],[17].

Supposons que G=(X,U) soit un graphe que nous désirons examiner. Au départ aucun sommet de G n'est encore examiné. Nous commençons la méthode à partir de n'importe quel sommet, et nous choisissons un arc que nous parcourons pour arriver à son extrémité terminale. Nous continuons de la sorte : à chaque pas, nous choisissons un arc dont l'extrémité initiale a déjà été atteinte et nous traversons cet arc. L'extrémité terminale de l'arc peut déjà avoir été atteinte. Si ce sommet n'a pas encore été atteint, nous continuons une nouvelle exploration à partir de ce point. Eventuellement, nous parcourons ainsi tous les arcs de G, chacun une et une seule fois. Cette méthode s'appelle une **recherche**.

Il y a beaucoup de façons de "rechercher" un graphe : chaque recherche dépend du critère de choix des arcs. Nous pouvons choisir un arc partant du sommet atteint le plus récemment, ce sommet constituant l'extrémité initiale d'un arc non encore parcouru. Une telle recherche est appelée **recherche en profondeur**.

Supposons maintenant que $\underline{G}$ soit un multigraphe. Comme une arête peut être considérée comme un arc pointant dans deux directions, nous pouvons appliquer une recherche à $\underline{G}$. Celle-ci oriente chaque arête de $\underline{G}$ selon la direction suivie pour traverser l'arête. Ainsi le multigraphe $\underline{G}$ devient un graphe P, c'est-à-dire si $\underline{G}=(X,\underline{U})$ où $X=\{x_1,x_2,\ldots,x_n\}$ et $\underline{U}=(\underline{u}_1,\underline{u}_2,\ldots,\underline{u}_n)$ avec $\underline{u}_i=(\underline{x}_i,\underline{x}_j)$ alors $P=(X,U)$ où $X=\{x_1,x_2,\ldots,x_n\}$ et $U=(u_1,u_2,\ldots,u_n)$ avec $u_i=(x_i,x_j)$ ou (x_j,x_i) selon le sens suivant lequel l'arête $(\underline{x}_i,\underline{x}_j)$ a été parcourue lors de la recherche. L'ensemble des arcs qui mènent à un sommet non encore examiné lors de la recherche définit une arborescence maximale de P. Ces arcs sont appelés les **branches** de P. En général, les arcs de P n'appartenant pas à l'arborescence (appelés **cordes**) interconnectent les chemins de l'arborescence engendrant, par là-même des cycles. Cependant, si la recherche est en profondeur, chaque corde (x_i,x_j) relie un sommet x_i à un de ses ascendants.

Définition

Soit P=(X,U) un graphe où $U=U_1 \cup U_2$.

Supposons que P vérifie les deux propriétés suivantes :

i) le graphe partiel $T=(X,U_1)$ est une arborescence maximale de P

ii) $\forall u \in U_2, \exists x_i,x_j$ tq ($u=(x_i,x_j)$ et $\exists \mu[x_j,x_i] \in T$)

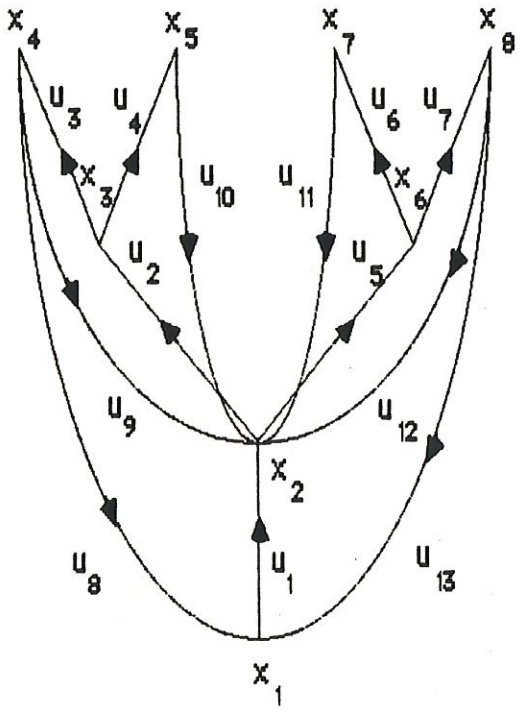
c'est-à-dire chaque corde relie un sommet avec un de ses ascendants dans T.

Si P vérifie ces deux conditions, alors P est appelé un **palmier**.

Remarque

U1 est l'ensemble des branches et U2 est l'ensemble des cordes.

Exemple



$X=\{x_1,\ldots,x_8\}$

$U_1=\{u_1,\ldots,u_7\}$

$U_2=\{u_8,\ldots,u_{13}\}$

T=(X,U) est une arborescence maximale de P puisque T est simplement connexe, sans cycle et admet une racine x_1. De plus, chaque corde (par exemple u_8) relie un sommet (x_4) avec un de ses ascendants dans T (x_1).

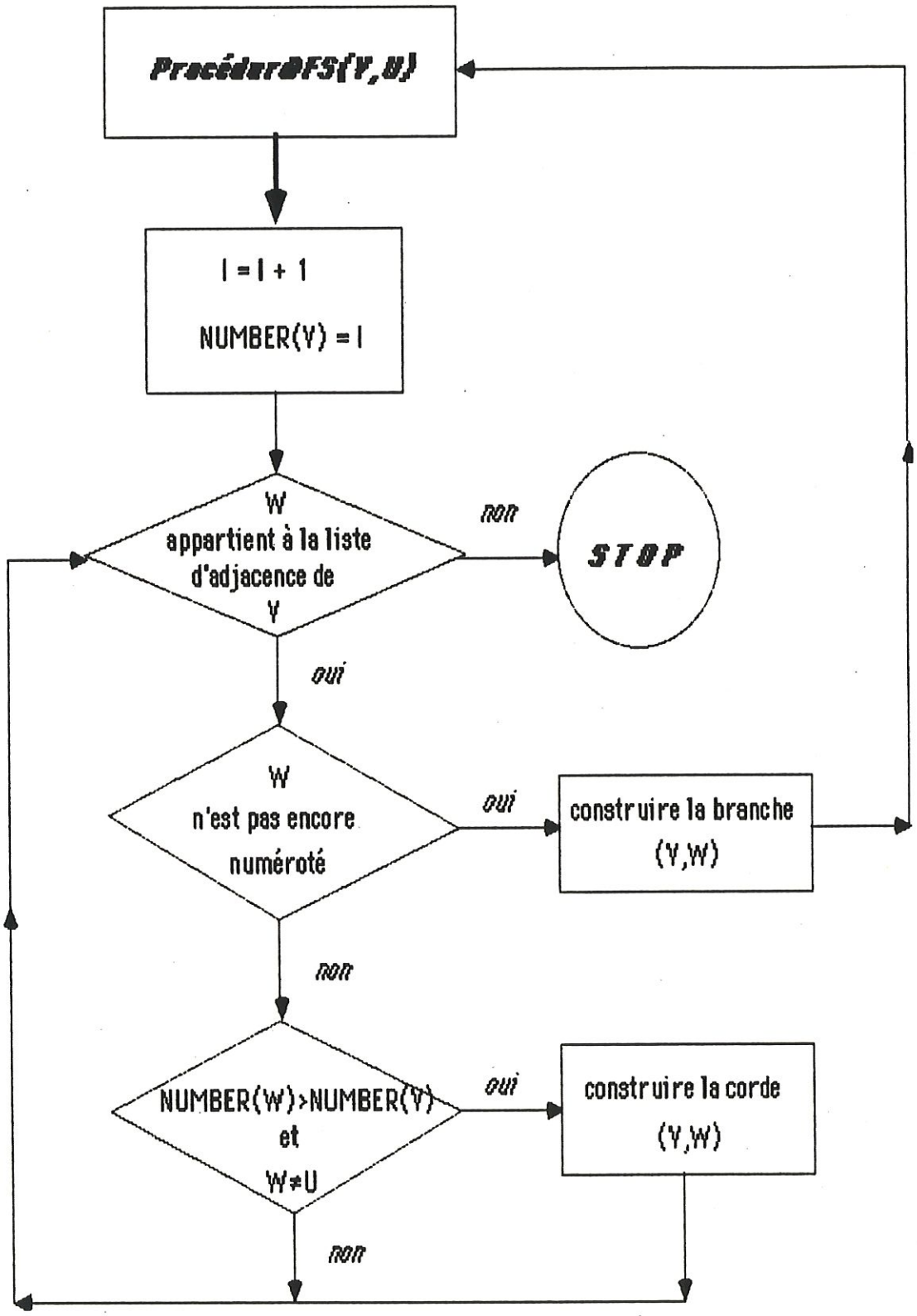
R. Tarjan a montré le résultat suivant :

Théorème

Si P est le graphe engendré par une recherche en profondeur d'un multigraphe simplement connexe G, alors P est un palmier.

Considérons le programme de la page suivante qui effectue une recherche en profondeur d'un multigraphe simplement connexe partant d'un sommet S et utilisant la structure d'adjacence du multigraphe. Le programme numérote les sommets du multigraphe dans l'ordre où ils sont atteints durant la recherche et construit le graphe P par la recherche.

I:=0
DFS(S,0)


Procédure DFS(V,U)
I = I + 1
NUMBER(V) = I
W appartient à la liste d'adjacence de V
non
STOP
oui
W n'est pas encore numéroté
oui
construire la branche (V,W)
non
NUMBER(W)>NUMBER(V) et W≠U
oui
construire la corde (V,W)
non

Avant de montrer que la recherche ainsi décrite engendre bien un palmier, essayons de voir ce qu'elle fait.

Supposons que nous soyons au sommet V et que nous venons d'examiner le sommet U. La procédure DFS (Depth-First Search) nous dit que le sommet atteint c'est-à-dire V, est numéroté afin de savoir que nous aurons déjà examiné ce sommet si ultérieurement nous le rencontrons encore. Remarquons que le numéro de V est supérieur d'une unité à celui du sommet U qui le précède.

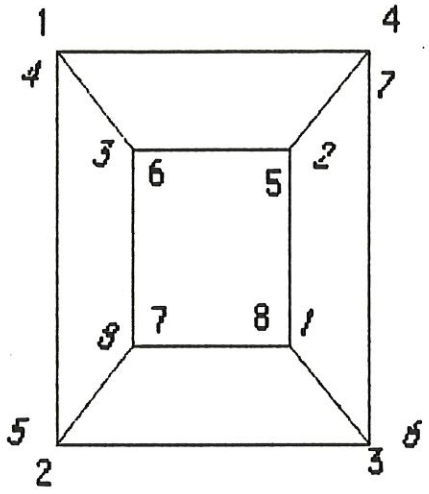
Ensuite, nous nous intéressons à un sommet W adjacent au sommet V. Un tel sommet existe puisque le multigraphe de départ G est simplement connexe. Trois cas peuvent alors se produire :

1e cas : le sommet W n'a pas encore été rencontré et n'est donc pas encore numéroté. Dès lors, nous construisons l'arc (V,W) qui devient une branche de T et nous appelons récursivement la procédure DFS à partir de ce nouveau sommet W;

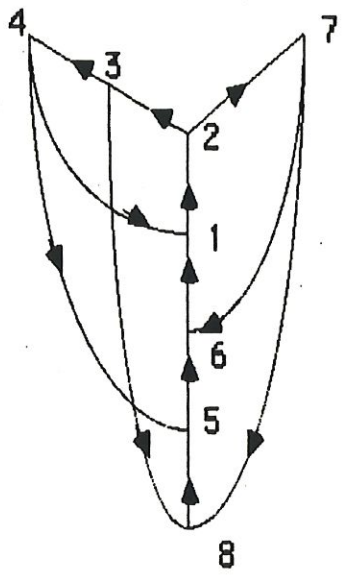
2e cas : le sommet W n'a pas encore été atteint et constitue un ascendant de V mais non le précédent de V. Dès lors, nous construisons l'arc (V,W) qui devient une corde de P et nous allons examiner un autre sommet adjacent à V si ce sommet existe. S'il n'existe pas, nous revenons au sommet U et nous poursuivons cette procédure avec un autre sommet adjacent que V, non encore examiné. Si ce sommet n'existe pas, nous revenons au sommet précédant U s'il existe et nous appliquons de nouveau cette procédure...

3e cas : le sommet W a déjà été atteint et est soit un descendant de V (mais non son suivant), soit le précédent de V. Ces deux cas n'apporte rien à la construction du palmier et on passe à un autre sommet adjacent au sommet V. Si ce sommet n'existe pas, comme dans le deuxième cas, nous revenons au sommet U ou à tout autre ascendant de V qui possède encore des sommets adjacents non encore examinés.

Pour mieux saisir l'action de l'algorithme, voici un multigraphe auquel nous appliquons une recherche en profondeur : les nombres écrits en italique représentent les nouveaux numéros des sommets, les couples représentent les arcs construits, et le schéma représente le palmier engendré par l'algorithme.



(8,5)
(5,6)
(6,1)
(1,2)
(2,3)
(3,8)
(3,4)
(4,1)
(4,5)
(2,7)
(7,6)
(7,8)



Montrons maintenant que, P=(X,U), le graphe engendré par la recherche en profondeur de $\underline{G}$=(X,$\underline{U}$) est un palmier.

Démonstration

L'algorithme ne boucle pas car d'une part tout multigraphe possède un nombre fini de sommets. Chacun de ses sommets admet donc un nombre fini de sommets adjacents. D'autre part, chaque sommet ne peut être numéroté qu'une fois : si W est déjà numéroté, alors on n'appelle pas la procédure DFS(W,V) où W aurait été de nouveau numéroté.

Tous les sommets sont numérotés. Par l'absurde, supposons qu'il existe un sommet W qui ne soit pas numéroté. Comme le multigraphe est simplement connexe, il existe un sommet $V \in X$ tel que W soit adjacent à V. Par hypothèse, V est numéroté. Puisque W est dans la liste d'adjacence de V et que W n'est pas numéroté, l'algorithme appelle la procédure DFS(W,V) qui numérotera W. Or W n'est pas numéroté. Dès lors, ou bien W n'est pas adjacent à V, ou bien W est numéroté. Comme W n'est pas numéroté (par hypothèse), il faut donc que W ne soit pas adjacent à V, ce qui entraîne une contradiction.

Par construction, le sommet S de départ n'a aucun précédent. Tout autre sommet V est l'extrémité terminale d'exactement un seul arc. Le graphe partiel $T=(X,U_1)$ est évidemment simplement connexe (sinon, cela signifierait que le multigraphe de départ $\underline{G}$ n'était pas simplement connexe, ce qui contredit l'hypothèse du théorème) et sans cycle (toute branche $u \in U_1$ est de la forme u=(V,W) où le numéro de V est toujours supérieur à celui de W). Par conséquent, T est une arborescence maximale de P.

Toute arête du multigraphe G est orientée. Si l'arête (V,W) n'est pas devenue une branche de l'arborescence T, alors soit (V,W), soit (W,V) est une corde puisque V et W sont numérotés lorsque l'arête (V,W) est examinée et que NUMBER(V)<NUMBER(W) ou NUMBER(V)>NUMBER(W).

Le numéro de l'extrémité initiale de chaque branche est inférieur au numéro de l'extrémité terminale. Le numéro de l'extrémité initiale de chaque corde est supérieur au numéro de l'extrémité terminale.

Si on construit la corde (V,W), alors la branche (W,V) ne peut pas être construite ultérieurement car V est numéroté. D'autre part, si la branche (W,V) est construite, alors on ne construira pas la corde (V,W) à cause du test "W≠U". Ainsi, chaque arête du multigraphe est orientée dans une et une seule direction.

Considérons une corde (V,W). Nous savons que NUMBER(W) < NUMBER(V). Cela signifie que le sommet W a été numéroté avant le sommet V. Puisque la corde (V,W) est construite et non la branche (W,V), le sommet V a dû être numéroté avant que l'arête (W,V) ne soit examinée. Ainsi, le sommet V a dû être numéroté durant l'exécution de DFS(W,.). Toutefois, tous les sommets numérotés durant l'exécution de DFS(W,.) sont des descendants du sommet W. Il en résulte que l'extrémité terminale de la corde est un ascendant du sommet V.

Cela achève de montrer que le graphe P=(X,U) engendré par la recherche en profondeur d'un multigraphe simplement connexe G est un palmier.

Notons que la réciproque du théorème est également vraie c'est-à-dire que tout palmier est engendré par une recherche en profondeur du multigraphe P=(X,U).

Algorithmique

La principale difficulté que nous avons rencontrée lors de l'implémentation est que la méthode de recherche en profondeur est récursive alors que le Fortran ne permet que des méthodes itératives. Pour pallier à cet inconvénient, nous avons utilisé une pile que nous notons PILE. Chaque couche de la pile contient trois renseignements :

- le sommet précédant le sommet que nous examinons. C'est le sommet noté U dans l'algorithme de Tarjan;
- le sommet que nous examinons. C'est le sommet V;
- un sommet W adjacent au sommet que nous examinons.

Désignons par N le pointeur vers la dernière couche remplie de la pile.

Stratégie

+ Initialisation :
  - Soit V le sommet que nous examinons. Au départ, nous choisissons arbitrairement ce sommet.
  - Soit U le sommet précédant le sommet V. Comme au départ, V est la racine de l'arborescence, il n'existe pas de sommet précédant V. Par convention, U est le sommet numéroté 0.
  - Nous recherchons ensuite un sommet W adjacent au sommet de départ V. Nous décidons de prendre le dernier sommet de la liste d'adjacence de V. Les autres sommets de cette liste d'adjacence sont mis dans la pile.
  - Soit K le nombre d'arcs du palmier déjà construits. Contrairement à l'algorithme de Tarjan, nous ne passons pas deux fois en revue toutes les arêtes : nous nous arrêtons dès que K est égal à NAR c'est-à-dire dès que nous avons construit tous les arcs du palmier. Initiallement, K vaut donc 0.
  - Enfin, le sommet V doit être numéroté à 1 c'est-à-dire NUMBER(V)=I(où I=1), alors que les autres sommets portent le numéro 0 c'est-à-dire $\forall$ J=1,...,NSO et J≠V, NUMBER(J)=0.

+ Boucle :

Désignons par PALM l'ensemble des couples de sommets définissant les arcs du palmier.

Tant que nous n'avons pas construit tous les arcs du palmier (c'est-à-dire pour tout les sommets K avec K<NAR) :

+ si le sommet W adjacent au sommet V n'a pas encore été numéroté, alors :
  - on numérote W (NUMBER(W)=I où I a été incrémenté d'une unité);
  - on construit la branche (V,W) :
    . on incrémente K,
    . PALM(1,K)=V et PALM(2,K)=W;
  - le sommet W devient le sommet courant :
    . U=V, V=W,
    . recherche du dernier sommet de la liste d'adjacence de V et mise sur pile des autres sommets adjacents à V;
+ sinon si le numéro de W est inférieur au numéro de V et si le sommet W n'est pas le précédent de V, alors
  - on construit la corde (V,W) :
    . on incrémente K,
    . PALM(1,K)=V et PALM(2,K)=W;
  - on met à jour les sommets U, V, W :
    . ces trois sommets constituent les trois éléments contenus dans la dernière couche de la pile,
    . on lit la dernière couche de la pile et on met à jour du pointeur N;
+ sinon, on met à jour les sommets U, V, W :
  - ces trois sommets constituent les trois éléments contenus dans la dernière couche de la pile,
  - on lit la dernière couche de la pile et on met à jour du pointeur N.

5.3.2.2 Vérification de la connexité

Rappelons-nous que nous devons voir si le multigraphe est simplement connexe. Que se passe-t-il si nous appliquons une recherche en profondeur à un multigraphe qui n'est pas simplement connexe ?

Puisqu'une recherche en profondeur engendre un palmier correspondant à un multigraphe simplement connexe, si le multigraphe de départ n'est pas simplement connexe, alors l'algorithme engendre un sous-palmier c'est-à-dire qu'il engendre le palmier correspondant à la composante simplement connexe contenant le sommet initial. Dès lors, seuls les sommets de cette composante simplement connexe seront numérotés. Par conséquence, nous voila munis d'un critère permettant de décider si le multigraphe est simplement connexe :

+ nous effectuons une recherche en profondeur;
+ nous vérifions qu'aucun sommet ne porte le numéro 0 (c'est-à-dire NUMBER(I)≠0, ∀ 1≤I≤NSO) pour pouvoir affirmer que le multigraphe est simplement connexe.

5.3.3 La molécule est un graphe simple

Nous devons vérifier que la molécule forme un 1-multigraphe sans boucle. Nous avons à notre disposition la structure d'adjacence du multigraphe (voir le paragraphe 5.3.1). Vérifier que le multigraphe est simple revient à examiner la liste d'adjacence de chaque sommet du multigraphe et à vérifier que la liste d'adjacence du $K^{ème}$ sommet (1≤K≤NSO) ne contient ni le sommet K (⇒ pas de boucle), ni deux sommets identiques (⇒ 1-multigraphe).

Stratégie

Pour la liste d'adjacence de chaque sommet K (1≤K≤NSO) :

+ si le sommet ADJL(1,K) est égal au sommet K, nous arrêtons et nous signalons l'erreur puisque nous avons trouvé une boucle. Remarquons que le sommet ADJL(1,K) existe toujours quelque soit K car le multigraphe est simplement connexe (ceci a fait l'objet du paragraphe 5.3.2).

+ sinon, tant qu'il existe encore un sommet dans la liste d'adjacence du sommet K :
  . nous vérifions que chaque sommet de la liste d'adjacence du sommet K est différent du sommet K (sinon nous arrêtons et nous signalons l'erreur)
  . nous vérifions que chaque sommet de la liste d'adjacence est différent des sommets qui le précèdent dans cette liste (sinon nous arrêtons et nous signalons l'erreur).

5.4 Vérifications concernant la liaison autour de laquelle une rotation est effectuée

5.4.1 La liaison doit exister

La liaison est définie par la connaissance de quatre atomes I, J, K et L. Ceux-ci sont les extrémités de trois liaisons : les liaisons (I,J), (J,K) et (K,L). La liaison (J,K) est celle autour de laquelle nous effectuons une rotation. Ces quatre atomes définissent en fait l'angle de torsion c'est-à-dire l'angle formé par les plans engendrés par les triplets (I,J,K) et (J,K,L) (voir le paragraphe 5.2.1). Nous voyons l'importance de ces trois liaisons dont il faut vérifier l'existence.

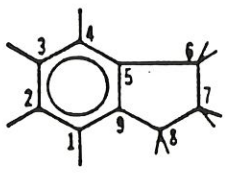
Algorithmique

Dans l'article publié par le laboratoire de Chimie moléculaire structurale et radiocristallographie (C.M.S.) des Facultés Universitaires N-D de la Paix de Namur [12], on teste l'existence de l'angle de torsion grâce à la condition suivante :

(D(I,J)=1 et D(I,K)=2 et D(I,L)=3)

où D est la matrice de distance de la molécule. Toutefois, les auteurs de l'article font remarquer que ce test peut échouer si la molécule possède des cycles.

Exemple



Si nous considérons les indices 5, 6, 7 et 8, D(5,8) défini par le plus court chemin n'est pas égal à trois liaisons. Il faut donc imaginer une autre méthode.

Stratégie

Nous avons à notre disposition la structure d'adjacence de la molécule. Pour chaque atome I, J et K, nous vérifions que les atomes J, K et L appartiennent respectivement à la liste d'adjacence de ces atomes. Si ce n'est pas le cas, alors l'angle de torsion n'est pas défini correctement.

5.4.2 La liaison doit être simple

Les liaisons doubles et triples sont chimiquement trop fortes pour permettre une rotation. Un fitting ne peut donc être effectué qu'autour d'une liaison simple.

Malheureusement, il n'existe aucune règle permettant de détecter à coup sûr la nature d'une liaison. Généralement, on peut prédire la nature d'une liaison grâce au type des atomes qui la définissent. Toutefois, il existe des exceptions qui rendent tout repérage impossible.

Nous avons donc dû choisir un compromis : nous utilisons le critère des types mais, après avoir averti l'utilisateur, nous lui laissons le soin de confirmer ou d'infirmer notre prédiction.

Algorithmique

L'utilisateur introduit généralement le type des atomes de la molécule afin de caractériser si ceux-ci peuvent réaliser une liaison multiple. Les atomes susceptibles de former une liaison double ou triple présentent la particularité d'orienter leurs orbitales suivant un agencement autre que tétraèdrique. Les types atomiques pouvant entraîner un liaison double ou triple sont : 2, 3, 4, 7, 8, 9, 11, 13 et 23.

Stratégie

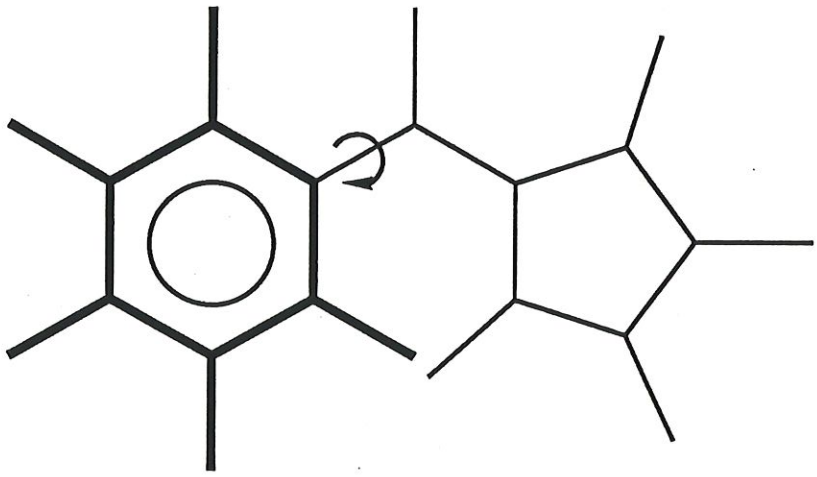
Une liaison peut ne pas être simple si le type de chacun des atomes la définissant est l'un des neuf types repris ci-dessus. Si c'est le cas pour la liaison autour de laquelle nous effectuons une rotation, nous avertissons l'utilisateur que la liaison est probablement double ou triple. Si il confirme notre prévision, la liaison n'est donc pas simple et nous ne pouvons pas effectuer une

rotation. Si il infirme notre prévision, la liaison est bien simple et nous pouvons continuer.

5.4.3 la liaison ne peut pas appartenir à un cycle

Etant donné une liaison chimique simple, le fitting fait tourner autour de cette liaison la partie mobile de la molécule tout en laissant inchangée la partie fixe de la molécule (voir le paragraphe 5.2.1).

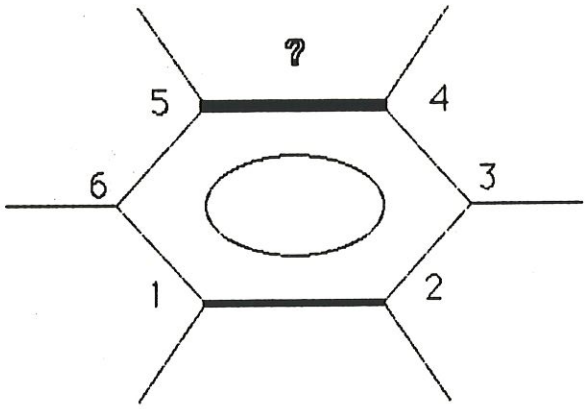
Exemple



On imagine dès lors sans peine que la partie mobile doit être disjointe de la partie fixe. Autrement dit, la liaison autour de laquelle nous effectuons le fitting ne peut appartenir à un cycle.

Exemple

Voir la page suivante.



Si nous utilisons le formalisme dans sa version actuelle :

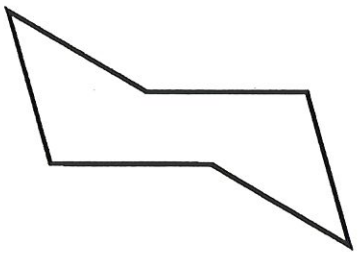
D(3,1)-D(3,2)=+1 D(4,1)-D(4,2)=+1
D(5,1)-D(5,2)=-1 D(6,1)-D(6,2)=-1

Nous remarquons donc que les atomes 3 et 4 appartiennent à la partie mobile, alors que les atomes 5 et 6 appartiennent à la partie fixe. Or il existe une liaison entre un atome de la partie fixe et un atome de la partie mobile : la liaison (4,5).

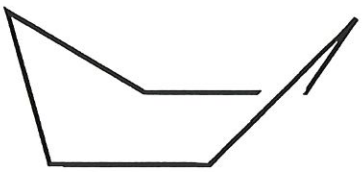
Remarque

Nous devons toutefois signaler que certains mouvements sont possibles à l'intérieur de certains cycles, appelés cycles saturés (de forme chaise ou bateau), mais cela dépasse le cadre de nos préoccupations.

Exemples



Molécule chaise



Molécule bateau

Nous devons donc vérifier que la liaison autour de laquelle nous effectuons une n'appartient à aucun cycle. Examinons les trois méthodes suivantes.

5.4.3.1 Première méthode

Une première méthode serait de parcourir toutes les chaînes possibles qui commencent par l'arête à tester (c'est-à-dire la liaison autour de laquelle nous effectuons une rotation), et de voir si on revient au sommet de départ. On arrêterait ce travail soit dès que l'on a trouvé un cycle, soit dès que l'on a passé en revue toutes les chaînes possibles.

Comme l'utilisateur est censé rentrer des données correctes, nous pouvons espérer que la liaison soit très souvent admissible. Cela a pour conséquence de rendre le premier critère inefficace. Nous sommes donc condamnés à passer en revue toutes les chaînes pour nous rendre compte que la liaison est (très souvent) correcte ! On se rend compte que cette méthode n'est pas très optimale.

Les deux méthodes suivantes font appel à la méthode de recherche en profondeur.

5.4.3.2 Deuxième méthode

A) *Reformulation du problème*

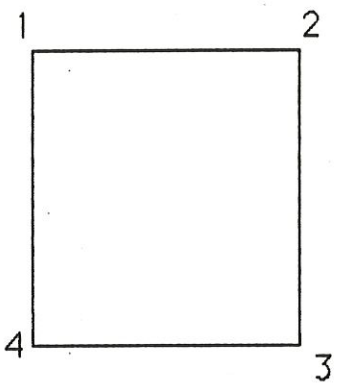
Voyons comment l'algorithme de recherche en profondeur permet de vérifier si une arête appartient, ou non, à un cycle. Soient $\underline{G}=(X,\underline{U})$ le multigraphe de départ, $\underline{u}=(\underline{x}_1,\underline{x}_2)$ l'arête à tester, $P=(X,U)$ où $U=U_1 \cup U_2$ le palmier engendré par la recherche.

Puisque P est le graphe engendré par le multigraphe $\underline{G}$ en orientant ses arêtes selon la recherche, tout circuit de P est un cycle de $\underline{G}$.

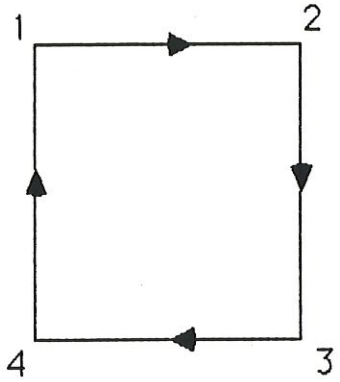
Exemple

Voir la page suivante.

Considérons le multigraphe



et le palmier engendré



Dès lors, vérifier que l'arête $(\underline{x}_1,\underline{x}_2)$ appartient à un cycle dans $\underline{G}$ revient à vérifier que l'arc (x_1,x_2) (ou (x_2,x_1)) appartient à un cycle dans P. Or P jouit d'une structure bien particulière : celle de palmier.

Nous savons par définition que si $P=(X,U_1 \cup U_2)$ est un palmier, alors P admet un graphe partiel $T=(X,U_1)$ qui est une arborescence maximale de P et les cordes de P relient un sommet avec un de ses ascendants dans T. Dès lors, cet ascendant devient aussi le suivant du sommet formant l'extrémité initiale de la corde. Autrement dit, la corde a créé un circuit dans le graphe P. Il nous reste à voir si les circuits ainsi créés contiennent l'arc (x_1,x_2) (ou l'arc (x_2,x_1)).

Nous savons que nous pouvons commencer la recherche en profondeur à partir de n'importe quel sommet. Toutefois, nous décidons de commencer l'algorithme à partir d'une des extrémités de l'arête à tester. Soit x_1 (par exemple) ce sommet. De plus, nous faisons en sorte que l'arête $(\underline{x}_1,\underline{x}_2)$ devienne le première branche de l'arborescence T. **Dès lors, si il existe une corde dont l'extrémité initiale est un descendant de x_2 dans T et l'extrémité terminale est x_1, il existe un circuit dans P contenant l'arc (x_1,x_2) et donc un cycle dans $\underline{G}$ contenant l'arête $(\underline{x}_1,\underline{x}_2)$.**

Remarques

- Nous aurions pu choisir le sommet x_2 comme racine de l'arborescence T et imposer que l'arc (x_2,x_1) soit la première branche de T.

- En fait, nous aurions pu choisir n'importe quel sommet du multigraphe, mais nous nous serions heurtés à des difficultés supplémentaires :

. l'arête $(\underline{x}_1,\underline{x}_2)$ du multigraphe peut devenir l'arc (x_1,x_2) ou l'arc (x_2,x_1). De plus, cet arc peut être une corde ou une branche. Nous nous trouvons donc devant quatre alternatives quant à la transformation de l'arête $(\underline{x}_1,\underline{x}_2)$;

. si l'arc (x_1,x_2) (ou (x_2,x_1)) forme une corde, alors cet arc fait partie d'un circuit puisque, par construction, toute corde engendre un circuit dans P. Ce cas est le plus facile, mais il ne faut pas se leurrer; rien ne garantit que l'arête $(\underline{x}_1,\underline{x}_2)$ deviendra une corde du palmier P;

. si l'arc (x_1,x_2) (ou (x_2,x_1)) forme une branche, alors nous devons vérifier qu'il existe une corde dont l'extrémité initiale est un descendant de x_2 (respectivement x_1) et dont l'extrémité terminale est un ascendant de x_1 (respectivement x_2) dans T.

Nous voyons donc que si nous choisissons x_1 comme sommet de départ, x_1 forme la racine de T et ne possède qu'un seul ascendant : le sommet x_1 lui-même. Cela nous évite donc de devoir rechercher les ascendants de x_1, ce qui représente un gain important.

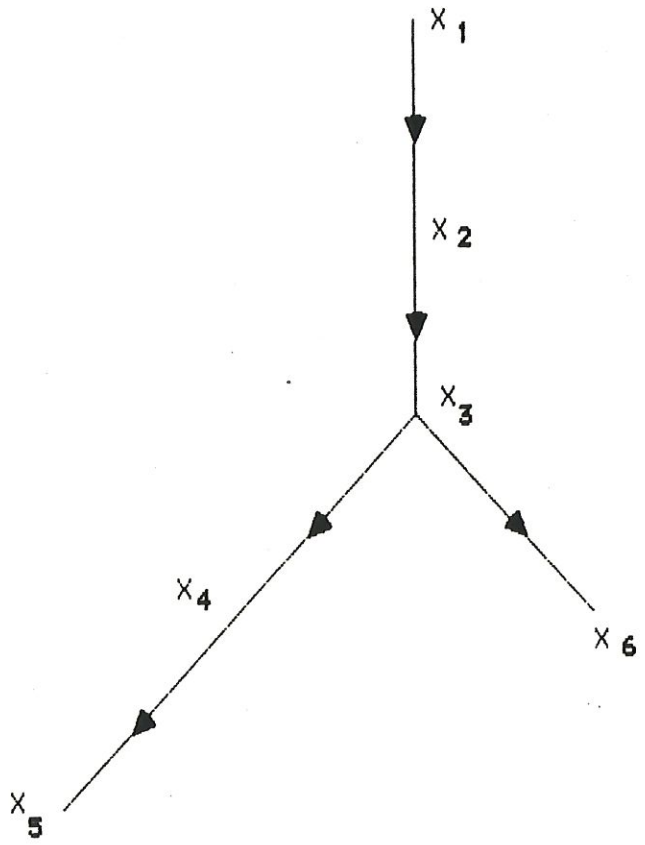
B) *Résolution du nouveau problème*

Nous devons donc vérifier qu'il existe une corde reliant un descendant de x_2 avec le sommet x_1. Rappelons-nous à l'aide de l'exemple ci-dessous, la manière dont la recherche en profondeur construit l'arborescence T.

Le graphe partiel T est construit à partir de sa racine x_1 de sorte que l'arc (x_1,x_2) forme la première branche de l'arborescence.

Ensuite, on "accroche" à (x_1,x_2) une branche dont l'extrémité initiale est x_2 et ainsi de suite jusqu'à ce que l'on parvienne à un sommet x_5 qui n'a plus de sommet adjacent non encore atteint. Dans ce cas, nous revenons à un ascendant x_3 de ce sommet, ascendant qui possède encore au moins un sommet adjacent non encore atteint x_6 et nous recommençons le processus à partir de ce sommet x_3.

Exemple



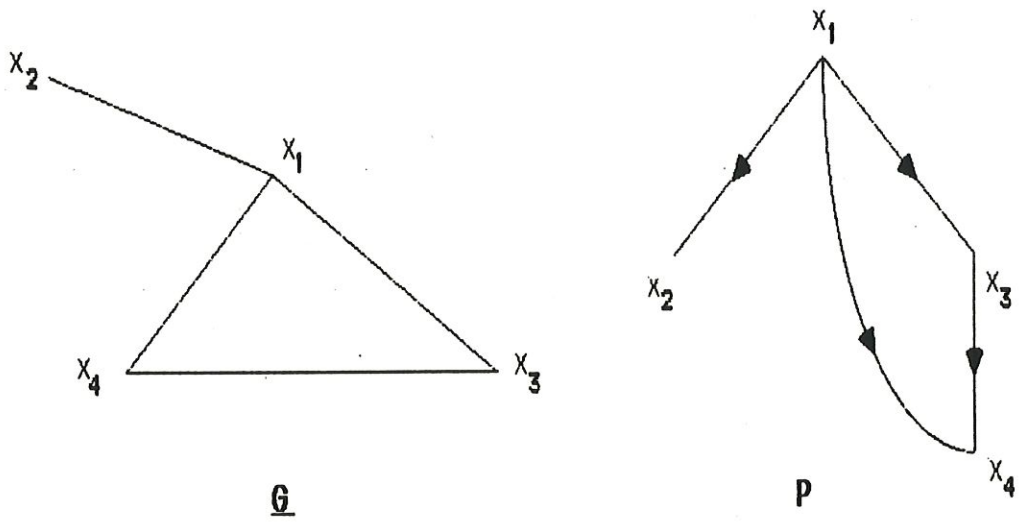
Il est important de bien comprendre ce processus de construction de T car il permet de ne pas devoir chercher explicitement les descendants de x_2. En effet, nous distinguons deux types de sommets dans $X\setminus\{x_1,x_2\}$:

- d'une part, les sommets que l'on peut atteindre dans T en partant de x_2;
- d'autre part, les sommets qui ne font pas partie d'un chemin partant de x_2.

La recherche en profondeur construit d'abord les branches qui

relient x_2 et ses descendants. Ensuite, elle revient à x_1 et elle construit les branches qui relient x_1 et ses descendants qui ne sont pas des des- cendants de x_2.

Exemple



Cette observation nous fournit une méthode permettant de vérifier si une arête $(\underline{x}_1,\underline{x}_2)$ appartient à un cycle.

Algorithmique

+ Effectuer une recherche en profondeur en commençant par le sommet x_1 et en choisissant l'arête $(\underline{x}_1,\underline{x}_2)$ comme première branche branche de l'arborescence.
+ Comme l'algorithme de recherche en profondeur commence la construction de P par les descendants de x_2, nous examinons la suite des arcs:
  - si l'extrémité terminale d'un arc est x_1, alors nous avons trouvé une corde reliant un des descendants de x_2 à x_1. Dès lors, l'arête $(\underline{x}_1,\underline{x}_2)$ appartient à un cycle;
  - si l'extrémité initiale d'un arc (sauf le premier) est la racine x_1, alors nous avons examiné tous les descendants de x_2 sans trouver de corde les reliant à x_1. Dans ce cas, nous pouvons affirmer que l'arête $(\underline{x}_1,\underline{x}_2)$ n'appartient pas à un cycle;
  - si tous les arcs ont été examinés sans que l'on ait trouvé un arc

dont l'extrémité initiale ou l'extrémité terminale soit x_1, alors l'arête (x_1,x_2) n'appartient pas à un cycle.

C) *Avantages et inconvénients*

Cette méthode est moins coûteuse car elle passe en revue chaque arête au maximum trois fois. Elle peut se décomposer en deux parties :

+ une première partie où le multigraphe est réordonné en palmier suite à une recherche en profondeur. Rappelons-nous que dans cette méthode, chaque arête est parcourue au plus deux fois;

+ une deuxième partie où nous passons en revue les branches et les cordes du palmier afin de vérifier si la liaison autour de laquelle nous effectuons une rotation appartient, ou non, à un cycle. On parcourt chaque arête au maximum une seule fois (certaines arêtes peuvent ne pas être examinées).

Toutefois, si cette méthode convient bien pour la vérification d'<u>une</u> arête, elle s'avère coûteuse si nous sommes amenés à effectuer plusieurs fois cette vérification. En effet, à chaque vérification nous engendrons un palmier. Dans la troisième méthode, nous montrons qu'il suffit d'engendrer un palmier une et une seule fois et de repérer grâce à ce graphe <u>toutes</u> les arêtes faisant partie d'un cycle.

5.4.3.3 Troisième méthode

Dans ce qui suit, nous exposons comment à partir du palmier engendré nous pouvons repérer définitivement toutes les arêtes appartenant à un cycle.

A) *Rappel*

Soit $\underline{G}=(X,\underline{U})$ le multigraphe, $P=(X,U)$ le palmier engendré par la recherche en profondeur effectuée à partir de $\underline{G}$, où $U=U_1 \cup U_2$ avec U_1 l'ensemble des branches, c'est-à-dire l'ensemble des arcs de l'arborescence $T=(X,U_1)$, et U_2 l'ensemble des cordes. Rappelons que, comme P est un palmier, chaque corde relie un sommet avec un de ses ascendants dans T formant ainsi un circuit dans P et par conséquent, un cycle dans $\underline{G}$.

B) *Méthode*

Puisque les cordes du palmier engendrent un circuit, nous allons repérer ces cordes. Ensuite, pour chacune de ces cordes, nous allons déterminer le cycle créé. Pour cela, nous allons parcourir le chemin reliant dans T l'extrémité terminale de la corde et aboutissant à son extrémité initiale. Les arcs de ce chemin correspondent aux arêtes du multigraphe appartenant au cycle engendré par la corde. Il suffit donc de partir de l'extrémité terminale de la corde, de choisir un de ses sommets suivants, de s'y rendre et de recommencer ce processus jusqu'à ce que nous atteignions l'extrémité initiale.

Le problème, c'est qu'il n'existe pas nécessairement un chemin unique partant de l'extrémité terminale de la corde et qu'il nous faut donc repérer parmi tous ces chemins celui passant par l'extrémité initiale de la corde. C'est précisément cet inconvénient que nous avons voulu éviter en rejetant la première méthode au profit de la recherche en profondeur. Cette difficulté nait du fait qu'un sommet peut avoir plusieurs descendants dans l'arborescence T. Par contre, chaque sommet ne possède qu'un seul précédent dans T (par construction). Dès lors, il nous suffit de partir de l'extrémité initiale de la corde, de "remonter" au précédent de ce sommet et de continuer de la sorte jusqu'à l'extrémité terminale de la corde.

Remarquons que nous sommes sûrs d'atteindre l'extrémité terminale car tout sommet n'a qu'un précédent dans T et que l'extrémité terminale est un ascendant de l'extrémité initiale dans T (P

est un palmier).

En appliquant cette méthode pour toutes les cordes du palmier, nous repérons toutes les arêtes appartenant à un cycle. En effet, la théorie des graphes nous dit que chaque corde engendre un cycle fondamental (ceux que nous venons de détecter). Or les cycles fondamentaux forment une base de l'espace vectoriel des sommes sans arête commune des cycles élémentaires [11]. Ainsi, tous les cycles du multigraphe sont engendrés par les cycles que nous avons repérés.

Remarque

La théorie des graphes nous apprend aussi qu'il y a $\nu(\underline{G})$ cycles fondamentaux où $\nu(\underline{G})$ est le nombre cyclomatique :

$\nu(G)=M-N+P$ où $M=\#U$ (nombre d'arêtes),

$N=\#X$ (ordre du multigraphe),

P= nombre de composantes simplement connexes (ici, P=1 puisque par hypothèse, le multigraphe est simplement connexe).

Exemple

Reprenons le multigraphe de la page **V.19** :
M=12, N=8, P=1 $\Rightarrow \nu(G)=12-8+1=5$.
Il y a donc cinq cordes dans P : (4,1), (4,5), (3,8), (7,6), (7,8).

C) *Algorithmique*

C1) *Problème principal*

Il ne suffit pas de repérer les arêtes du multigraphe qui appartiennent à un cycle : nous devons nous donner un moyen permettant de "retenir" définitivement quelles sont ces arêtes. Pour cela, envisageons la **matrice des cycles** C du multigraphe $\underline{G}$:
$\forall\, x_i, x_j,\ 1\le x_i,x_j\le NSO$, $C(x_i,x_j)=0$ si et seulement si l'arête $(\underline{x_i},\underline{x_j})$ n'appartient à aucun cycle du multigraphe, $C(x_i,x_j)=1$ si et seulement si l'arête $(\underline{x_i},\underline{x_j})$ appartient à un cycle du multigraphe. Comme nous travaillons avec des multigraphes simples, $C(x_i,x_i)=0\ \ \forall x_i,\ 1\le x_i\le NSO$.

Par conséquent, dès que l'on a créé cette matrice des cycles C, si nous voulons vérifier que l'arête $(\underline{x}_i,\underline{x}_j)$ appartient ou non à un cycle, il suffit de voir si $C(x_i,x_j)$ vaut respectivement 1 ou 0.

Remarque

La matrice des cycles d'un multigraphe est symétrique.

Voyons maintenant comment contruire cette matrice C. Pour cela, nous avons à notre disposition le palmier P engendré lors de la vérification du caractère simplement connexe de la molécule (lire le paragraphe 5.3.2). Remarquons que contrairement à la deuxième méthode, nous ne devons imposer ni la racine de l'arborescence T, ni la première branche de T.

Stratégie

+ Nous initialisons la matrice C : $\forall$ I,J $\leq$ NSO, $1 \leq I,J \leq NSO$, C(I,J)=0.
+ Nous examinons tous les arcs du palmier; soit (x_1,x_2) l'arc courant :
  - si (x_1,x_2) est une branche, nous passons à l'arc suivant;
  - si (x_1,x_2) est une corde (c'est-à-dire si $NUMBER(x_1) > NUMBER(x_2)$),
    . cet arc fait partie d'un circuit $\Rightarrow C(x_1,x_2)=1$ et $C(x_2,x_1)=1$;
    . nous "remontons" l'arborescence T à partir de x_1 jusqu'à x_2 en marquant les arêtes rencontrées (c'est-à-dire en mettant à jour la matrice des cycles C);
    . nous passons à l'arc suivant.

Nous sommes donc amenés à remonter l'arborescence T en marquant les arêtes rencontrées. Avant de détailler l'algorithmique de ce sous-problème, remarquons que nous avons peut-être déjà repéré des arêtes appartenant à un cycle. Dans ce cas, si la chaîne reliant x_1 à x_2 est formée de certaines de ces arêtes, il est inutile de les marquer de nouveau.

C'est pourquoi nous avons créé la matrice ligne PREC; PREC(I) désigne le sommet précédant le sommet I dans l'arborescence T. A l'origine, tous les éléments de cette matrice sont nuls. Nous modifions cette matrice au fur et à mesure que nous parcourons les arêtes reliant dans l'arborescence l'extrémité initiale d'une corde à l'extrémité terminale de cette corde. PREC permet ainsi de retrouver les chaînes formées d'arêtes appartenant à un cycle.

Par conséquent, si PREC(x_1) est nul, le sommet x_1 ne fait partie d'aucune des éventuelles chaînes déjà examinées. Dans ce cas, nous "remontons" l'arborescence à partir du sommet x_1 jusqu'au sommet x_2 en marquant les arêtes non déjà marquées de la chaîne.

Par contre, si PREC(x_1) n'est pas nul, il existe une chaîne reliant un descendant A de x_1 (A peut être égal à x_1) à un ascendant B de x_1. Deux cas peuvent se produire :

- le sommet B est un ascendant du sommet x_2 ou le sommet x_2 lui-même;
- le sommet B est un descendant de x_2.

Dans le premier cas, nous avons déjà examiné la chaîne $\mu[A,B]$ où $\mu[x_1,x_2] \subset \mu[A,B]$. Comme toutes les arêtes de la chaîne $\mu[A,B]$ ont déjà été marquées, il est inutile de marquer les arêtes de la chaîne $\mu[x_1,x_2]$. Dans le deuxième cas, seule une partie de la chaîne $\mu[x_1,x_2]$ a déjà été marquée : il s'agit de la chaîne $\mu[x_1,B]$. Nous devons encore marquer la chaîne $\mu[B,x_2]$ (cette chaîne existe puisque B est un descendant de x_2 et $B \neq x_2$). En résumé, si PREC$(1,x_1)$ n'est pas nul, nous devons rechercher le sommet B et si ce sommet B est un descendant de x_2 ($B \neq x_2$), alors nous marquons les arêtes non déjà marquées de la chaîne $\mu[B,x_2]$.

Remarque

Nous "remontons" dans des circonstances analogues, deux fois l'arborescence en marquant les arêtes non déjà marquées de la chaîne :

- la première fois, nous marquons les arêtes de la chaîne $\mu[x_1,x_2]$ où x_2 est ascendant de x_1 (différent de x_1) avec PREC(x_1)=0;
- la deuxième fois, nous marquons les arêtes de la chaîne $\mu[B,x_2]$ où x_2 est ascendant de B ($B \neq x_2$) avec PREC(B)=0. En effet, si PREC(B)≠0, alors B n'est pas l'extrémité de la chaîne partant de A, ce qui est contraire aux hypothèses.

C2) *Sous-problèmes*

Il nous reste à traiter deux sous-problèmes; d'une part, comment rechercher l'extrémité d'une chaîne composée d'arêtes déjà marquées, et d'autre part, comment "remonter" l'arborescence en marquant les arêtes non déjà marquées.

Remonter l'arborescence

Soit PALM l'ensemble des couples de sommets définissant les arcs du palmier P construit par la recherche en profondeur.

Nous allons parcourir la chaîne de l'arborescence T partant du sommet PALM(1,N) et aboutissant à un de ses ascendants Y dans T. Toutes les arêtes de cette chaîne font partie d'un cycle. Nous marquerons donc ces arêtes (si elles ne sont pas déjà marquées). En vertu de la remarque que nous avons faite à la page précédente, nous supposons que la première arête de cette chaîne n'a pas encore été marquée (c'est-à-dire PREC(PALM(1,N))=0).

Stratégie :

+ Nous initialisons le sommet courant X au sommet de départ, c'est-à-dire que X=PALM(1,N).
+ Tant que nous n'avons pas atteint le sommet Y, c'est-à-dire que tant que le sommet courant X est un descendant du sommet Y dans T (NUMBER(X) > NUMBER(Y)), nous examinons l'arc qui précède l'arc N dans PALM :
  - si l'extrémité terminale x_2 de cet arc (x_2=PALM(1,N') où N'=N-1) n'est pas le sommet X, alors cet arc n'est pas adjacent à ce sommet. Nous examinons alors l'arc précédant cet arc dans PALM;
  - sinon, si l'extrémité terminale x_2 représente le sommet X, alors l'arc (x_1,x_2) est adjacent au sommet X (où x_1=PALM(1,N')). Par conséquent, le sommet x_1 précède le sommet X dans l'arborescence T :
    . le sommet x_1 est donc le précédent du sommet X le long d'une chaîne formée d'arêtes appartenant à un cycle. Dès lors, nous devons compléter la matrice PREC : le $x^{ème}$ élément de PREC représente le sommet qui précède le sommet X
    $\Rightarrow$ PREC(X)=x_1;

. nous devons poursuivre ce processus : le sommet x_1 devient le nouveau sommet courant. Contrairement à l'ancien sommet X (PALM(1,N)), le sommet x_1 peut être l'extrémité d'une chaîne précédemment parcourue (c'est le cas si PREC(X)≠0). Il existe alors une chaîne $\mu[x_1,x_2]$ reliant le sommet courant x_1 à un de ses sommets ascendants y_1, cette chaîne étant composée d'arêtes déjà marquées. Comme nous l'avons déjà exposé précédemment, nous nous trouvons devant deux cas :

. le sommet y_1 est un ascendant de Y ou est le sommet Y,
. le sommet y_1 est un descendant de Y.

Dans le premier cas, la chaîne $\mu[x_1,Y] \subset \mu[x_1,y_1]$. Comme toutes les arêtes de la chaîne $\mu[x_1,y_1]$ ont déjà été marquées et que l'arête $(\underline{X},\underline{x}_1)$ vient d'être marquée, toutes les arêtes de la chaîne $\mu[X,Y]$ ont été marquées. Nous pouvons donc arrêter.
Dans le deuxième cas, seule une partie de la chaîne $\mu[x_1,Y]$ a déjà été marquée : il s'agit de la chaîne $\mu[x_1,y_1]$. Il nous reste donc à marquer la chaîne $\mu[y_1,Y]$. Cette chaîne existe puisque y_1 est descendant de Y et $Y \neq y_1$.
Pour satisfaire ces deux cas, il suffit que le sommet courant devienne le sommet y_1. En effet, dans le premier cas, y_1 est ascendant de Y (ou y_1=Y), c'est-à-dire NUMBER(y_1)≤ NUMBER(Y), ce qui constitue la négation de la condition gérant la boucle. Nous sortons donc de la boucle et nous arrêtons bel et bien. Par contre, dans le deuxième cas, NUMBER(y_1)>NUMBER(Y), ce qui satisfait la condition de boucle. Nous entrons donc de nouveau dans la boucle avec y_1 comme sommet courant c'est-à-dire que nous parcourons la chaîne $\mu[y_1,Y]$.

Rechercher l'extrémité de la chaîne

Enfin, de manière générale, nous devons rechercher dans T l'ascendant d'un sommet S qui constitue l'extrémité d'une chaîne passant par S, chaîne dont les arêtes font partie d'un cycle.

Stratégie :

Ce problème se résoud facilement grâce à la matrice PREC. En effet, comme le sommet S appartient à une chaine dont les arêtes font partie d'un cycle, PREC(S) représente le précédent de S le long de cette chaîne. Soit X ce sommet précédent. Deux cas peuvent se produire :

- PREC(X)=0 : cela signifie que le sommet X est l'extrémité de la chaîne puiqu'il n'a pas de précédent le long de cette chaîne. Nous avons donc trouvé l'ascendant recherché;
- PREC(X)≠0 : dans ce cas, le sommet X n'est pas l'extrémité de la chaîne puisque X possède un précédent le long de la chaîne. Dans ce cas, nous regardons si ce précédent possède, ou non, un précédent et nous continuons ainsi ce processus.

Remarque

Quoique le sommet trouvé ne possède pas de précédent le long de la chaîne parcourue, il est très possible que ce sommet ne soit pas la racine de l'arborescence et possède donc un précédent dans T.

Chapitre VI

Utilisation du programme

Manuel d'utilisation du programme reprenant les quatre méthodes de comparaison moléculaire.

6.1 Utilisation générale

Démarrage du logiciel

Le programme principal se trouve dans le fichier FITTIN.FOR.

Pour exécuter le programme, il vous suffit donc de taper au terminal : RUN FITTIN.

Titre

Apparaît sur l'écran le titre du programme.

Celui-ci reste affiché durant 15 secondes.

```
**********************************************************************
*                                                                    *
*                          E F M F I T                               *
*                          -----------                               *
*                                                                    *
*                  Techniques de comparaison moleculaire             *
*    tenant compte de l'energie de contrainte de la molecule mobile ou*
*    permettant de considerer plusieurs liaisons comme axe de rotation.*
*                                                                    *
*                                                                    *
*                                                                    *
```

Un message d'erreur apparaît sur l'écran, si dans les données d'une molécule:
- il existe une liaison entre un atome et lui-même,
- un atome est lié à plus de quatre autres atomes,
- il existe deux liaisons identiques,
- la connexité de la molécule n'est pas correcte.

Vous devez alors réintroduire la molécule.

```
+-----------------------------+
|      Entree des donnees     |
+-----------------------------+

Introduisez le nom du fichier contenant la molecule fixe.
NOM : fixe2
Introduisez le nom du fichier contenant la molecule mobile.
NOM : mobile1
Introduisez le nom du fichier contenant la molecule mobile.
NOM : mobile2
```

On vous demande alors d'entrer le nom du fichier sur lequel seront écrits les différents résultats du fitting flexible que vous allez effectuer (se rapporter au chapitre 7 pour l'organisation de ce fichier).

```
+-----------------------------+
|      Entree des donnees     |
+-----------------------------+

Introduisez le nom du fichier contenant la molecule fixe.
NOM : fixe
Introduisez le nom du fichier contenant la molecule mobile.
NOM : mobile
Introduisez le nom du fichier sur lequel seront inscrites les tierces.
NOM : resultats
```

Remarques:

1. Il faut préciser le suffixe de votre fichier si celui-ci n'est pas un fichier ***.dat.
2. Il n'a pas été prévu un traitement des erreurs pouvant intervenir lors de la lecture-écriture d'un fichier.

Si vos données sont mal écrites ou si le fichier n'existe pas, le programme peut donc soit mal interpréter les données, soit s'arrêter.

Dans ce dernier cas, l'ordinateur affiche un message d'erreur qui vous permet de résoudre plus facilement le problème.

Choix du fitting

S'affiche le menu principal du programme.

Celui-ci vous propose les différentes techniques de comparaison moléculaire existant sur ce logiciel :

- fitting sur plusieurs axes de rotation,

  Il permet d'utiliser plusieurs liaisons simples comme axes de rotation.(chap IV)

- fitting avec contrainte d'énergie,

  Celui-ci calcule la solution optimale pour laquelle l'énergie ne dépasse pas un seuil de 20 kcal/mole au-dessus du minimum d'énergie de conformation (chap 3.2.b).

- fitting pour une énergie admissible au départ,

  Celui-ci recherche l'optimum pour lequel l'énergie ne dépasse pas le seuil cité ci-dessus et pour lequel de l'angle de départ à cet angle optimal, l'énergie ne passe pas par un maximum trop élevé (chap 3.2.c).

- fitting parcourant l'énergie.

  Celui-ci recherche l'optimum en modélisant les minima et maxima d'énergie pour s'assurer qu'il n'existe pas entre l'angle optimal et l'angle de départ un maximum trop élevé (chap 3.3).

Choisissez un de ces fittings flexibles.

```
+-----------------------------+
|      Choix du fitting       |
+-----------------------------+

Vous pouvez choisir un fitting :
                1. sur plusieurs axes de rotation.
                2. avec contraintes d'energie.
                3. pour une energie admissible en zero.
                4. en parcourant l'energie.

Votre choix (1..4) : 2
```

En sortant de ce fitting flexible.

Fin d'itération

On vous propose d'effectuer un autre fitting flexible (retour au choix du fitting) ou d'arrêter l'exécution du programme.

```
+-----------------------------+
|      Fin d'iteration        |
+-----------------------------+

Desirez-vous effectuer un nouveau fitting sur ces deux molecules ?
Veuillez repondre par "OUI" ou "NON" : non

Desirez-vous effectuer un AUTRE fitting sur ces deux molecules ?
Veuillez repondre par "OUI" ou "NON" : non
```

Fin du programme

Paraît la sortie de la molécule mobile.

On vous demande le nom du fichier sur lequel sera sauvée la nouvelle molécule mobile, résultat des fittings flexibles que vous venez d'effectuer.

Remarque :

Il n'est pas obligatoire de préciser le suffixe du fichier si celui-ci est un fichier ***.dat.

```
+-----------------------------+
|      Fin du programme       |
+-----------------------------+

Introduisez le nom du fichier de sortie de la molecule mobile.
NOM : sortie1
```

6.2 Fitting sur plusieurs axes de rotation

Itération :

Nous vous demandons d'entrer les données relatives à une itération du fitting.

C'est-à-dire le nombre d'atomes à superposer,
les atomes de la molécule fixe à superposer,
les atomes de la molécule mobile à superposer,
le nombre de liaisons simples prises comme axe de rotation (ce nombre doit être inférieur ou égal au nombre d'atomes à superposer),
les liaisons formant les axes de rotation.

```
                    Iteration  1
Introduisez les atomes a comparer et les axes.

Entrez le nombre d'atomes a comparer : 3
Entrez les  3 atomes a egaler de la molecule fixe :   5 6 7
Entrez les  3 atomes a egaler de la molecule mobile : 5 6 7
Entrez le nombre d'axes : 3
Entrez le  1 eme axe : 2 3 4 5
Entrez le  2 eme axe : 3 4 5 6
Entrez le  3 eme axe : 4 5 6 7
```

Un message d'erreur apparaît sur l'écran si la liaison prise comme axe de rotation:
- n'existe pas,
- appartient à un cycle,
- la laison n'est pas simple.

Vous avez la possibilité de continuer si l'erreur est la dernière citée.

Dans les autres cas, réintroduisez les données.

```
.....ATTENTION.....
!Le nombre d'axes de rotation est plus grand que le nombre d'atomes a comparer.
.....
  Introduisez les atomes a comparer et les axes.

  Entrez le nombre d'atomes a comparer : 1
  Entrez les  1 atomes a egaler de la molecule fixe :   5
  Entrez les  1 atomes a egaler de la molecule mobile : 5
  Entrez le nombre d'axes : 1
  Entrez le  1 eme axe : 2 3 4 5
```

Minimisation :

On vous demande d'entrer pour chaque axe de rotation, un angle (en degré). Celui-ci permet d'initialiser la recherche des angles optimaux, résultats du fitting flexible.

Nous vous demandons d'entrer, si possible, des valeurs proches de la solution optimale.

```
+-------------------------+
|      Minimisation       |
+-------------------------+

Entrez une approximation de chaque angle de rotation :

    Angle( 1) autour de l'axe  3 -  4 = 60.
    Angle( 2) autour de l'axe  4 -  5 = 60.
    Angle( 3) autour de l'axe  5 -  6 = 60.
```

Viennent s'ajouter sur l'écran, les valeurs des angles optimaux ainsi que le rms.

Un message prévient l'uilisateur si le programme suppose que la solution n'est pas optimale.

```
+-------------------------+
|      Minimisation       |
+-------------------------+

Entrez une approximation de chaque angle de rotation :

    Angle( 1) autour de l'axe  3 -  4 = 60.
    Angle( 2) autour de l'axe  4 -  5 = 60.
    Angle( 3) autour de l'axe  5 -  6 = 60.

Solutions de la minimisation :

    179.9999359 autour de l'axe  3 -  4
    -179.999979 autour de l'axe  4 -  5
    -179.999979 autour de l'axe  5 -  6
    RMS =       0.0000000
```

Fin d'itération

On vous propose d'appliquer un nouveau fitting flexible (retour au menu itération) ou d'arrêter (retour au menu fin d'itération de l'utilisation générale).

```
+---------------------------+
|     Fin d'iteration       |
+---------------------------+

Desirez-vous effectuer un nouveau fitting sur ces deux molecules ?
Veuillez repondre par "OUI" ou "NON" : non
```

6.3 Fitting avec contrainte d'énergie

Itération

Nous vous demandons d'entrer les données relatives à une itération du fitting.

C'est-à-dire le nombre d'atomes à superposer,
les atomes de la molécule fixe à superposer,
les atomes de la molécule mobile à superposer,
le nombre de liaisons simples prises comme axe de rotation (ce nombre doit être inférieur ou égal au nombre d'atomes à superposer),
les liaisons formant les axes de rotation.

```
+-------------------------+
|      Iteration  1       |
+-------------------------+
Introduisez les atomes a comparer et l'axe.

Entrez le nombre d'atomes a comparer : 1
Entrez les  1 atomes a egaler de la molecule fixe :  5
Entrez les  1 atomes a egaler de la molecule mobile : 5
Entrez l'axe de rotation : 2 3 4 5
```

Un message d'erreur apparaît sur l'écran si la liaison prise comme axe de rotation:
- n'existe pas,
- appartient à un cycle
- la laison n'est pas simple.

Vous avez la possibilité de continuer si l'erreur est la dernière citée.

Dans les autres cas, réintroduisez les données.

```
.....................................ATTENTION.....................................
:    Il semble que la liaison autour de laquelle on effectue le FITTING        :
:    soit une liaison double ou triple ! ! !                                   :
:                                                                              :
................................................................................

Est-ce reellement le cas ?      non
```

Minimisation

1) Energie

On vous demande d'entrer un angle (en degré). Celui-ci permet d'initialiser la recherche de l'angle optimal minimisant l'*énergie de conformation*

Nous vous demandons d'entrer, si possible, un angle proche de la solution optimale.

```
                    Minimisation
ntrez une approximation de l'angle minimisant l'energie :

   Angle = 60.
```

Apparaît alors sur l'écran, l'angle optimal ainsi que la valeur optimale de l'énergie.

Un message prévient l'utilisateur si le programme suppose que la solution n'est pas optimale.

```
                    Minimisation
Entrez une approximation de l'angle minimisant l'energie :

    Angle = 60.
Solution de la minimisation de l'energie :

    Angle =      90.2995928
    Valeur de l'energie =       1.9191904
```

2) Distance

Vous entrez un angle (en degré). Celui-ci permet d'initialiser la recherche de l'angle optimal permettant la superposition des deux molécules.

Nous vous demandons d'entrer, si possible, un angle proche de la solution optimale.

```
                    Minimisation
Entrez une approximation de l'angle minimisant l'energie :

     Angle = 80.

Solution de la minimisation de l'energie :

     Angle =      80.2995926
     Valeur de l'energie =       1.8191904
Entrez une approximation de l'angle optimal recherche :

     Angle autour de l'axe  3 -  4 = 100.
```

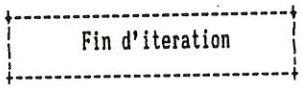
Apparaît alors sur l'écran, la valeur de l'angle optimal ainsi que le rms.

Un message prévient l'utilisateur si le programme suppose que la solution n'est pas optimale.

```
                    Minimisation
Entrez une approximation de l'angle minimisant l'energie :

     Angle = 80.

Solution de la minimisation de l'energie :

     Angle =      80.2995926
     Valeur de l'energie =       1.8191904
Entrez une approximation de l'angle optimal recherche :

     Angle autour de l'axe  3 -  4 = 100.
Solution avec contrainte d'energie :

        122.8099033 autour de l'axe  3 -  4
        RMS =       1.3873535
```

Fin d'itération

On vous propose d'appliquer un nouveau fitting flexible (retour au menu itération) ou d'arrêter l'exécution du programme (retour au menu fin d'itération de l'utilisation générale).




```
Desirez-vous effectuer un nouveau fitting sur ces deux molecules ?
Veuillez repondre par "OUI" ou "NON" : non
```

6.4 Fitting pour une énergie admissible en zero

Itération :

Nous vous demandons d'entrer les données relatives à une itération du fitting.

C'est-à-dire le nombre d'atomes à superposer,
les atomes de la molécule fixe à superposer,
les atomes de la molécule mobile à superposer,
la liaison formant l'axe de rotation.

```
+----------------------------+
|       Iteration  1         |
+----------------------------+
Introduisez les atomes a comparer et l'axe.

Entrez le nombre d'atomes a comparer : 1
Entrez les  1 atomes a egaler de la molecule fixe :  5
Entrez les  1 atomes a egaler de la molecule mobile : 5
Entrez l'axe de rotation : 2 3 4 5
```

Un message d'erreur apparaît sur l'écran si la liaison prise comme axe de rotation:
- n'existe pas,
- appartient à un cycle
- la laison n'est pas simple.

Vous avez la possibilité de continuer si l'erreur est la dernière citée.

Dans les autres cas, réintroduisez les données.

```
.....................................ATTENTION.....................................
:     La liaison autour de laquelle on effectue le FITTING n'existe pas      :
.....................................................................................
  Introduisez les atomes a comparer et l'axe.

  Entrez le nombre d'atomes a comparer : 1
  Entrez les  1 atomes a egaler de la molecule fixe :  5
  Entrez les  1 atomes a egaler de la molecule mobile : 5
  Entrez l'axe de rotation : 2 3 4 5
```

Minimisation

1) Energie

On vous demande d'entrer un angle (en degré). Celui-ci permet d'initialiser la recherche du minimum d'énergie.

Nous vous demandons d'entrer, si possible, un angle proche de la solution optimale.

```
                    Minimisation
Entrez une approximation de l'angle minimisant l'energie :

    Angle = 60.
```

Apparaît alors sur l'écran, la valeur de l'angle optimal ainsi que la valeur optimale de l'énergie.

Un message prévient l'utilisateur lorsque le programme suppose que la solution n'est pas optimale.

```
                    Minimisation
Entrez une approximation de l'angle minimisant l'energie :

    Angle = 60.
Solution de la minimisation de l'energie :

    Angle =      60.2995926
    Valeur de l'energie =      1.8191304
```

2) Distance

Apparaît alors l'angle optimal permettant de superposer les deux molécules, ainsi que le rms.

```
                    +---------------------------+
                    |       Minimisation        |
                    +---------------------------+
Entrez une approximation de l'angle minimisant l'energie :

     Angle = 60.
Solution de la minimisation de l'energie :

     Angle =      80.2995926
     Valeur de l'energie =       1.8191304
Solution sans contrainte d'energie :

     180.0000050 autour de l'axe  3 -  4
     RMS =        0.0000100
Solution sous contrainte d'energie :

    -122.9210528 autour de l'axe  3 -  4
     RMS =        1.3879505
```

Fin d'itération

On vous propose d'appliquer un nouveau fitting flexible (retour au menu itération) ou d'arrêter l'exécution du fitting (retour au menu fin d'itération de l'utilisation générale).

```
                    +---------------------------+
                    |      Fin d'iteration      |
                    +---------------------------+

Desirez-vous effectuer un nouveau fitting sur ces deux molecules ?
Veuillez repondre par "OUI" ou "NON" : oui
```

6.5 Fitting parcourant l'énergie

Itération :

Nous vous demandons d'entrer les données relatives à une itération du fitting.

C'est-à-dire le nombre d'atomes à superposer,
les atomes de la molécule fixe à superposer,
les atomes de la molécule mobile à superposer,
la liaison formant l'axe de rotation.

```
                    Iteration  2

Introduisez les atomes a comparer et l'axe.

Entrez le nombre d'atomes a comparer : 1
Entrez les  1 atomes a egaler de la molecule fixe :  6
Entrez les  1 atomes a egaler de la molecule mobile : 6
Entrez l'axe de rotation : 3 4 5 6
```

Un message d'erreur apparaît sur l'écran si la liaison prise comme axe de rotation:
- n'existe pas,
- appartient à un cycle
- la laison n'est pas simple.

Vous avez la possibilité de continuer si l'erreur est la dernière citée.

Dans les autres cas, réintroduisez les données.

```
.....................................ATTENTION.....................................
: La liaison autour de laquelle on effectue le FITTING appartient a un cycle  :
.....................................................................................

   Introduisez les atomes a comparer et l'axe.

   Entrez le nombre d'atomes a comparer : 1
   Entrez les  1 atomes a egaler de la molecule fixe :  5
   Entrez les  1 atomes a egaler de la molecule mobile : 5
   Entrez l'axe de rotation : 1 2 3 4
```

Minimisation

Apparaît sur l'écran, la valeur de l'angle optimal permettant la superposition des deux molécules, ainsi que la valeur du rms.

```
                    Minimisation

Solution sans contrainte d'energie :
   -122.3905767 autour de l'axe  4 -  5
   RMS =          2.0142250

Solution sous contrainte d'energie :
   -122.3905767 autour de l'axe  4 -  5
   RMS =          2.0142330
```

Fin d'itération

On vous propose d'appliquer un nouveau fitting flexible (retour au menu itération) ou d'arrêter (retour au menu fin d'itération de l'utilisation générale).

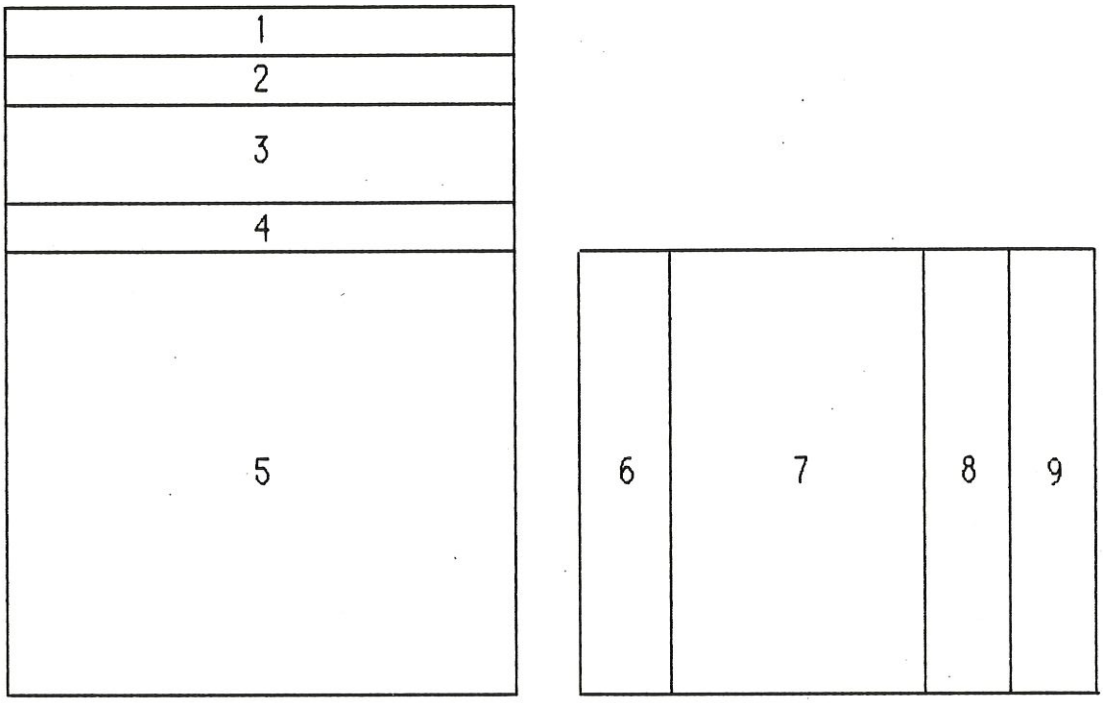
```
                    Fin d'iteration

Desirez-vous effectuer un nouveau fitting sur ces deux molecules ?
Veuillez repondre par "OUI" ou "NON" : oui
```

Chapitre VII

Organisation des fichiers

Dans ce chapitre, nous présentons la structure des fichiers des données et des fichiers des résultats.

7.1 Structure des fichiers des données



1. Première ligne : nom de la molécule ou titre du fichier.

2. Deuxième ligne : nombre de liaisons de la molécule.

3. Ensemble des paires d'atomes définissant les liaisons.

4. Nombre d'atomes de la molécule (sur une ligne).

5. Ensemble des informations relatives à un atome de la molécule :

   6. nombres atomiques;

   7. coordonnées;

   8. types;

   9. charges atomiques brutes sans les électrons de cœur.

7.2 Structure des fichiers des résultats

..... molécules à comparer .....

N°	Molécule fixe	Molécule mobile
1	2	3

..... Itération ... .

* Données de l'itération :				
N°	Molécule fixe		Molécule mobile	
4	5		6	
Les atomes à comparer :				
7	8	9	10	11
12				
Les axes de rotation :				
13	14	15	16	X
17				

* Les résultats de l'itération :				
18	19	20	21	22
23				
Les angles de rotation :				
24				
Le RMS :				
25				

..... Résultat final .....

N°	Molécule fixe	Molécule mobile
26	27	28

Molécules à comparer

1. Numéros désignant les atomes.
2. Coordonnées des atomes de la molécule fixe.
3. Coordonnées des atomes de la molécule mobile.

Données de l'itération

4. Numéros désignant les atomes.

5. Coordonnées des atomes de la molécule fixe.

6. Coordonnées des atomes de la molécule mobile.

7. Numéros des atomes à comparer de la molécule fixe.

8. Coordonnées des atomes à comparer de la molécule fixe.

9. Numéros des atomes à comparer de la molécule mobile.

10. Coordonnées des atomes à comparer de la molécule mobile.

11. Distance entre les atomes à comparer de la molécule fixe et de la molécule mobile.

12. Valeur de la fonction objectif.

13. Numéros des atomes définissant une des deux extrémités de la liaison simple servant d'axe de rotation.

14. Coordonnées de ces atomes.

15. Numéros des atomes définissant la deuxième extrémité de la liaison simple servant d'axe de rotation.

16. Coordonnées de ces atomes.

17. Valeurs des angles de rotation servant à initialiser la minimisation.

Résultats de l'itération

18. Numéros des atomes à comparer de la molécule fixe.

19. Coordonnées des atomes à comparer de la molécule fixe.

20. Numéros des atomes à comparer de la molécule mobile.

21. Coordonnées des atomes à comparer de la molécule mobile.

22. Distance entre les atomes à comparer de la molécule fixe et de la molécule mobile.

23. Valeur de la fonction objectif.

24. Angles optimaux associés à chaque axe de rotation.

25. Racine carrée de "la valeur de la fonction objectif divisée par le nombre d'atomes à comparer".

Résultat final

26. Numéros désignant les atomes.

27. Coordonnées des atomes de la molécule fixe.

28. Coordonnées des atomes de la molécule mobile.

Chapitre VIII

Variables et procédures

Nous décrivons dans ce chapitre toutes les procédures et variables utilisées dans le programme.

8.1 Principaux paramètres

NDIM1 : nombre maximum d'atomes.
NDIM2 : nombre maximum de liaisons; NDIM2=2xNDIM1.
NAXE : nombre maximum d'axe de rotation.

8.2 Variables globales

Common /**adjacence**/ : contient la structure d'adjacence et des informations relatives au palmier.

ADJL(0:4,ndim1) : double précision; structure d'adjacence de la molécule mobile. ADJL(0,k) est le nombre de sommets adjacents au sommet k; ADJL(i,k) est le sommet adjacent au sommet k, 0 sinon (voir la fonction logique VERDEG).

PALM(2,ndim2) : entier; ensemble des couples de sommets définissant les arcs du palmier engengré par la recherche en profondeur (voir la procédure PALMIER).

NUM(ndim1) : entier; numérotation des sommets engendrée par la recherche en profondeur (voir la procédure PALMIER).

Common /**cycle**/ : contient des informations permettant de savoir si une liaison appartient à un cycle.

ARETE(ndim1,ndim1) : logique; matrice des cycles : ARETE(i,j) est "vrai" si la liaison formée par les atomes i et j appartient à un cycle, "faux" sinon.

Ce common est créé par les procédures REPCY et CHAINE.

Common /**dista**/ : contient la matrice de distance de la molécule mobile.

DIST(ndim1,ndim1) : entier; matrice de distance de la molécule mobile.

Ce common est créé par la procédure NODE

Common /**divisi**/ : contient des renseignements sur les atomes à comparer (fitting sur un axe).

NBATEG : entier; nombre d'atomes à comparer.
AME(ndim1) : entier; atomes de la molécule mobile à comparer.
AFE(ndim1) : entier; atomes de la molécule fixe à comparer.

Ce common est créé par la procédure LECDONBIS.

Common /**divisi**/ : contient des renseignements sur les atomes à comparer (fitting sur plusieurs axes).

NBATEG : entier; nombre d'atomes à comparer.
AME(0:ndim1,0:naxe) : entier; AME(.,0) est l'ensemble des atomes à superposer et AME(.,i) est l'ensemble des atomes à comparer se trouvant dans la partie mobile par rapport au $i^{ème}$ axe (voir la procédure MOBAT).
AFE(ndim1) : entier; atomes de la molécule fixe à comparer.

Ce common est créé dans les procédures LECDON et MOBAT.

Common /**energie**/ : contient les informations relatives à l'énergie de conformation de la molécule mobile.

EPS(ndim1,ndim1) : double précision; paramètres d'énergie de Van der Waals (voir la procédure INIVDW).
RVDW(ndim1) : double précision; rayons de Van der Waals (voir la procédure INIVDW).
QC(ndim1) : double précision; charges brutes des atomes (voir la procédure INICOUL).
VT(9) : double précision; constante de torsion (voir la procédure INITOR).
ST(9) : entier; ST(i) est le signe de NT(i) (voir la procédure INITOR).
NT(9) : double précision; période de la liaison (voir la procédure INITOR).

Common /**fixe**/ : contient les informations relatives à la molécule fixe.

XAT : entier; nombre d'atomes de la molécule fixe.
X(ndim1,3) : double précision; ensemble des atomes de la molécule fixe où X(i,j) est la $j^{ème}$ coordonnée du $i^{ème}$ atome.
XLIEN : entier; nombre de laisons de la molécule fixe.

Ce common est créé par la procédure DONNEE.

Common /**liaiso**/ : concerne la liaison autour de laquelle nous effectuons le fitting (fitting sur un axe).

AXE(4) : entier; désigne les quatre atomes définissant la liaison (voir la procédure LECDONBIS).
ATOMOB(0:ndim1) : entier; ATOMOB(0) représente le nombre d'éléments du tableau ATOMOB c'est-à-dire le nombre d'atomes de la molécule mobile qui bougent lors de la rotation (voir la procédure MOBATBIS).

Common /**liaiso**/ : concerne les liaisons autour desquelles nous effectuons le fitting (fitting sur plusieurs axes).

NBRAXE : entier; nombre d'axes de rotation (voir la procédure LECDON).

AXE(4,naxe) : entier; ensemble de quadruplets d'atomes définissant les axes de rotation (voir la procédure LECDON).

ATOMOB(0:ndim1,naxe) : entier; ensemble des atomes de la molécule mobile qui bougent lors de la rotation autour de chaque axe (voir la procédure MOBAT).

Common /**mobile**/ : contient les informations relatives à la molécule mobile.

YAT : entier; nombre d'atomes de la molécule mobile.

Y(ndim1,3) : double précision; ensemble des atomes de la molécule mobile où Y(i,j) est la $j^{ème}$ coordonnée du $i^{ème}$ atome.

YLIEN : entier; nombre de liaisons de la molécule mobile.

Ce common est créé par la procédure DONNEE.

Common /**molecu**/ : contient les informations relatives aux atomes de la molécule mobile.

BOND(2,ndim2) : entier; ensemble des paires d'atomes définissant les liaisons de la molécule.

NBAT(ndim1) : entier; ensemble des nombres atomiques des atomes.

ITYP(ndim1) : entier; ensemble des types des atomes.

CHGE(ndim1) : double précision; ensemble des charges brutes sans les électrons de cœur des atomes.

TITRE : caractère*70; titre de la molécule ou du fichier.

Ce common est créé par la procédure DONNEE.

Common /**precedent**/ : contient des informations permettant de "remonter" les chaînes formées par des arêtes appartenant à un cycle.

PREC(2,ndim1) : entier; PREC(1,k) : pointeur vers le sommet précédant le sommet k dans l'arborescence du palmier engendré par la recherche en profondeur, 0 sinon; PREC(2,k) : pointeur vers le couple de sommets définissant l'arc (PREC(1,k),k) du palmier.

Ce common est créé par la procédure CHAINE.

Common /**un**/ : contient un renseignement nécessaire pour le clacul de la fonction double précision Y.

A : double précision; permet de calculer la constante E(A)-20 de la fonction Y.

Ce common est créé par la procédure RECOPT.

Common /**tranf**/ : contient des renseignements relatifs aux translations des molécules.

CD(3) : double précision; cosinus directeurs de l'axe de rotation (voir la procédure COSDIR).

T(3) : double précision; vecteur de translation (voir les procédures TRANS et TRANSF).

8.3 Procédures et fonctions

Fonction logique **APRT** (IND,ATOMOB)

Description :

Cette fonction teste l'appartenance de l'atome IND à la partie mobile de la molécule.

Variables :

Paramètres :

+ IND : entier; numéro de l'atome.
+ ATOMOB(0:ndim1) : entier; ATOMOB(0) représente le nombre d'éléments du tableau ATOMOB c'est-à-dire le nombre d'atomes de la molécule qui bougent lors de la rotation.

Cette fonction est développée dans le fichier GEOMETRIE.FOR .

Procédure **CHAINE** (Y,K)

Description :

Cette procédure parcourt la chaîne de l'arborescence du palmier PALM (voir le common ADJACENCE) partant du sommet PALM(1,K) et aboutissant au sommet Y. Toutes les arêtes de cette chaîne fontalors partie d'un cycle. Nous mettons donc à jour les tableaux ARETE (voir common CYCLE) et PREC (voir common PRECEDENT). Pour plus de détails, se référer au paragraphe 5.4.3.3.

Variables :

Paramètres d'entrée :

+ Y : entier; extrémité de la chaîne.
+ K : entier; pointeur vers l'élément du tableau PALM représentant la corde qui engendre le cycle que nous examinons.

Variables internes :

+ X : entier; sommet courant.
+ X1 : entier; extrémité initiale de l'arc que nous examinons.
+ X2 : entier; extrémité terminale de l'arc que nous examinons.
+ N : entier; pointeur vers l'arc courant du tableau PALM.

Cette procédure est développée dans le fichier TESTAD.FOR .

Procédure **CONT** (DELTA,ETOR,EVDW,ECOUL)

Description :

Cette procédure calcule les énergies de torsion, de Van Der Waals et de Coulomb de la molécule mobile qui après une rotation autour d'une liaison simple.

Variables :

Paramètre d'entrée :

+ DELTA : double précision; angle de rotation exprimé en radian.

Paramètres de sortie :

+ ETOR : double précision; énergie de torsion de la molécule après la rotation d'un angle DELTA.
+ EVDW : double précision; énergie de Van Der Waals de la molécule après la rotation d'un angle DELTA.
+ ECOUL : double précision; énergie de Coulomb de la molécule après la rotation d'un angle DELTA.

Variable interne :

+ RACAR : double précision; carré de la distance euclidienne entre deux atomes.

Cette procédure est développée dans le fichier CALCUL.FOR .

Procédure **CONT1** (DELTA,ETOR1,EVDW1,ECOUL1)

Description :

Cette procédure calcule la valeur de la dérivée première des énergies de torsion, de Van Der Waals et de Coulomb de la molécule mobile qui après une rotation autour d'une liaison simple.

Variables :

Paramètre d'entrée :

+ DELTA : double précision; angle de rotation exprimé en radian.

Paramètres de sortie :

+ ETOR1 : double précision; valeur de la dérivée première de la fonction d'énergie de torsion de la molécule après la rotation d'un angle DELTA.

+ EVDW1 : double précision; valeur de la dérivée première de la fonction d'énergie de Van Der Waals de la molécule après la rotation d'un angle DELTA.
+ ECOUL1: double précision; valeur de la dérivée première de la fonction d'énergie de Coulomb de la molécule après la rotation d'un angle DELTA.

Variables internes :
+ RACAR : double précision; carré de la distance euclidienne entre deux atomes.
+ CCS1, CCS2 : double précision; constantes associées aux termes en cosinus de la fonction d'énergie.
+ CSN : double précision; constante associée au terme en sinus de la fonction d'énergie.
+ DERA1 : double précision; dérivée première de RACAR.

Cette procédure est développée dans le fichier CALCUL.FOR .

Procédure **CONT2** (DELTA,ETOR2,EVDW2,ECOUL2)

Description :

Cette procédure calcule la valeur de la dérivée seconde des énergies de torsion, de Van Der Waals et de Coulomb de la molécule mobile qui a subi une rotation autour d'une liaison simple.

Variables :

Paramètre d'entrée :
+ DELTA : double précision; angle de rotation exprimé en radian

Paramètres de sortie :
+ ETOR2 : double précision; valeur de la dérivée seconde de la fonction d'énergie de torsion de la molécule après la rotation d'un angle DELTA.
+ EVDW2 : double précision; valeur de la dérivée seconde de la fonction d'énergie de Van Der Waals de la molécule après la rotation d'un angle DELTA.
+ ECOUL2: double précision; valeur de la dérivée seconde de la fonction d'énergie de Coulomb de la molécule après la rotation d'un angle DELTA.

Variables internes :

+ RACAR : double précision; carré de la distance euclidienne entre deux atomes.
+ CCS1, CCS2 : double précision; constantes associées aux termes en cosinus de la fonction d'énergie.
+ CSN : double précision; constante associée au terme en sinus de la fonction d'énergie.
+ DERA1 : double précision; dérivée première de RACAR.
+ DERA2 : double précision; dérivée seconde de RACAR.

Cette procédure est développée dans le fichier CALCUL.FOR .

Procédure **COSDIR** (MAT,AXE,CD)

Description :

Cette procédure calcule les cosinus directeurs de l'axe de rotation.

Variables :

Paramètres d'entrée :

+ AXE(4) : entier; AXE(2) et AXE(3) sont les atomes qui définissent la liaison simple servant d'axe de rotation.
+ MAT(ndim1,3) : double précision; molécule mobile.

Paramètre de sortie :

+ CD(3) : double précision; cosinus directeurs.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Fonction double précision **DEG** (DELTA)

Description :

Cette fonction convertit en degré un angle exprimé en radian.

Variable :

Paramètre :

+ DELTA : double précision; angle à convertir en degré.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **DONNEE** (MOL,MAT,NAT,NLIEN)

Description :

Cette procédure lit un fichier de données contenant des renseignements sur une molécule : le nombre d'atomes, le nombre de liaisons, l'ensemble des paires d'atomes définissant les liaisons, l'ensemble des coordonnées de chaque atome, le nombre atomique, le type et la charge de chaque atome.

Variables :

Paramètre d'entrée :

+ MOL : caractère; contient les mots "fixe" ou "mobile" selon que nous lisons la molécule deréférence ou mobile.

Paramètres de sortie :

+ NAT : entier; nombre d'atomes.
+ NLIEN: entier; nombre de liaisons.
+ MAT(ndim1,3) : double précision; ensemble des coordonnées de chaque atome.

Variables internes :

+ NOM : caractère; nom du fichier des données.
+ TEST : logique; vérifie si les données sont introduites correctement.

Cette procédure est développée dans le fichier ENTREE.FOR .

Fonction double précision **E** (THETA)

Description :

Cette fonction prend la valeur de l'énergie pour un angle THETA.

Variables :

Paramètre d'entrée :

+ THETA : double précision; angle exprimé en degré.

Variable internes :

+ THETA1 : double précision; angle THETA exprimé en radian.

Cette fonction est développée dans le fichier MINFIT3.FOR .

Procédure **ECRITU**

Description :

Cette procédure écrit la molécule mobile après les fitting sur un fichier.

Variable :

Variable interne:

+ NOM : caractère; nom du fichier de résultat.

Cette procédure est développée dans le fichier SORTIE.FOR .

Fonction double précision **F** (THETA)

Description :

Cette fonction prend la valeur de la fonction objectif pour un angle THETA.

Variables :

Paramètre :

+ THETA : double précision; angle exprimé en degré.

Variable interne :

+ THETA1 : double précision; angle THETA exprimé en radian.

Cette fonction est développée dans le fichier MINFIT3.FOR .

Procédure **FITENE1** (ITER)

Description :

Cette procédure minimise la somme des carrés des distances entre les atomes à comparer sous contrainte d'énergie selon la méthode expliquée au paragraphe 3.2.B .

Variables :

Paramètre de sortie:

+ ITER : entier; nombre de fitting effectués.

Variable interne :

+ ENCORE : logique; "vrai" si l'utilisateur désire effecter un nouveau fitting flexible.

Cette procédure est développée dans le fichier FITENE1.FOR

Procédure **FITENE2** (ITER)

Description :

Cette procédure minimise la somme des carrés des distances entre les atomes à comparer sous contrainte d'énergie selon la méthode expliquée au paragraphe 3.2.C .

Variables :

Paramètre de sortie:

+ ITER : entier; nombre de fitting effectués.

Variable interne :

+ ENCORE : logique; "vrai" si l'utilisateur désire effecter un nouveau fitting flexible.

Cette procédure est développée dans le fichier FITENE2.FOR.

Procédure **FITENE3** (ITER)

Description :

Cette procédure minimise la somme des carrés des distances entre les atomes à comparer sous contrainte d'énergie selon la deuxième approche décrite au paragraphe 3.3 .

Variables :

Paramètre de sortie:

+ ITER : entier; nombre de fitting effectués.

Variable interne :

+ ENCORE : logique; "vrai" si l'utilisateur désire effecter un nouveau fitting flexible.

Cette procédure est développée dans le fichier FITENE3.FOR .

Procédure **FITMUL** (ITER)

Description :

Cette procédure effectue un fitting tenant compte de plusieurs liaisons comme axes de rotation.

Variables :

Paramètre de sortie :

+ ITER : entier; nombre de fitting effectués.

Variable interne :

+ ENCORE : logique; "vrai" si l'utilisateur désire effecter un nouveau fitting flexible.

Cette procédure est développée dans le fichier FITMUL.FOR .

Programme **FITTIN**

Description :

Programme principal.

Variable

Variable interne :

+ ENCORE : logique; "vrai" si l'utilisateur désire effecter un nouveau fitting flexible.

Ce programme est développé dans le fichier FITTIN.FOR .

Procédure **FONCT** (NFCT,NVAR,DELTA,FVEC,IFLAG)

Description :

Cette procédure évalue les valeurs de la somme des carrés des distances entre les atomes à comparer (rotations autour de plusieurs axes).

Variables :

Paramètres d'entrée :

+ NVAR : entier; nombre d'axes de rotation.
+ NFCT : entier; nombre d'atomes à comparer.

Paramètres de sortie :

+ DELTA(naxe) : double précision; angles de rotation optimaux associés à chacun des axes de rotation.
+ FVEC(3*naxe) : double précision; FVEC(i) est le carré de la distance séparant le $i^{ème}$ atome à comparer de la molécule fixe et le $i^{ème}$ atome à comparer de la molécule mobile.

Cette procédure est développée dans le fichier MINMUL.FOR .

Fonction double précision **G** (DELTA)

Description :

Cette fonction prend la valeur de la dérivée première de la fonction d'énergie calculée pour un angle DELTA (en degré).

Variables :

Paramètre :

+ DELTA : double précision; angle dont on recherche l'image par la fonction G.

Variable interne:

+ DELTA1 : double précision; angle DELTA exprimé en radian.

Cette fonction est développée dans le fichier MINFIT3.FOR .

Procédure **IMPATO**

Description :

Cette procédure imprime les coordonnées des atomes des molécules fixe et mobile sur le fichier des itérées.

Cette procédure est développée dans le fichier SORTIE.FOR .

Procédure **IMPEGA** (TEST,FNORM)

Description :

Cette procédure imprime les atomes à comparer (plusieurs axes de rotation) et la somme de carrés des distances sur le fichier des itérées. De plus, elle calcule le r.m.s.

Variables :

Paramètres d'entrée :

+ TEST : logique; si TEST est "vrai", nous soulignons le titre.
+ FNORM : double précision; r.m.s. c'est-à-dire la racine carrée de "la valeur optimale de la fonction objectif divisée par le nombre d'atomes à égaler".

Cette procédure est développée dans le fichier SORTIE.FOR .

Procédure **IMPEGABIS** (TEST,FNORM)

Description :

Cette procédure imprime les atomes à comparer (un seul axe de rotation) et la somme de carrés des distances sur le fichier des itérées. De plus, elle calcule le r.m.s.

Variables :

Paramètres d'entrée :

+ TEST : logique; si TEST est "vrai", mous soulignons le titre.
+ FNORM : double précision; r.m.s. c'est-à-dire la racine carrée de "la valeur optimale de la fonction objectif divisée par le nombre d'atomes à égaler".

Cette procédure est développée dans le fichier SORTIE.FOR .

Procédure **INFCT** (IDP,CCS,CSN)

Description :

Cette procédure calcule les constantes de la fonction objectif (voir le paragraphe 2.3.C).

Variables :

Paramètres de sortie :

+ IDP : double précision; terme indépendant de la fonction objectif.
+ CCS : double précision; constante associée aux termes en cosinus.
+ CSN : double précision; constante associée au terme en sinus.

Cette procédure est développée dans le fichier CALCUL.FOR .

Procédure **INICOUL** (YAT,NBAT,CHGE,QC)

Description :

Cette procédure initialise les paramètres de l'énergie de Coulomb.

Variables :

Paramètres d'entrée :

+ YAT : entier; nombre d'atomes.
+ NBAT(ndim1) : entier; nombre atomique de chaque atome.
+ CHGE(ndim1) : double précision; charges brutes sans les électrons de cœur.

Paramètre de sortie :

+ QC(ndim1) : double précision; charges brutes.

Cette procédure est développée dans le fichier ENTREE.FOR .

Procédure **INITOR** (AXE,ITYP,VT,NT,ST)

Description :

Cette procédure initialise les paramètres de l'énergie de torsion en fonction de l'axe de rotation.

Variables :

Paramètres d'entrée :

+ AXE(4) : entier; atomes définissant l'axe de rotation et l'angle de torsion.
+ ITYP(ndim1) : entier; type de chaque atome.

Paramètres de sortie :

+ VT(9) : double précision: barrière de torsion de chaque angle de torsion associé à l'axe de rotation défini par les atomes AXE(2) et AXE(3).
+ NT(9) : double précision; paramètre de torsion de chaque angle de torsion associé à l'axe de rotation défini par les atomes AXE(2) et AXE(3).
+ ST(9) : entier; ST(i) est le signe de NT(i).

Variables internes :

+ TYP1(dim), TYP2(dim) : entier; respectivement la première et la deuxième colonne du fichier contenant les paramètres de torsion : types des atomes déterminant les paramètres.
+ VD(dim) : double précision; troisième colonne du fichier contenant les paramètres de torsion : VD(i) est la barrière de torsion associée aux types TYP1(i) et TYP2(i).
+ FD(dim) : double précision; quatrième colonne du fichier contenant les paramètres de torsion : FD(i) est le paramètre de torsion associé aux types TYP1(i) et TYP2(i).

Cette procédure est développée dans le fichier ENTREE.FOR .

Procédure **INIVDW** (NAT,NLIEN,NBAT,BOND,RVDW,EPS)

Description :

Cette procédure initialise les paramètres de l'énergie de Van Der Waals.

Variables :

Paramètres d'entrée :

+ NAT : entier; nombre d'atomes.
+ NLIEN : entier; nombre de liaisons.
+ NBAT(ndim1) : entier; nombre atomique de chaque atome.
+ BOND(2,ndim2) : double précision; ensemble des paires d'atomes définissant les liaisons.

Paramètres de sortie :

+ RVDW(ndim1) : double précision; RVDW(i) est le rayon de Van Der Waals du i^{eme} atome.
+ EPS(ndim1,ndim1) : double précision; EPS(i,j) est la constante de Van Der Waals associée aux atomes i et j.

Variables internes:

+ E(10,10) : double précision; ensemble des constantes de Van Der Waals.
+ RS(10) : double précision; ensemble des rayons de Van Der Waals.

Cette procédure est développée dans le fichier ENTREE.FOR .

Procédure **LECDON**

Description :

Cette procédure lit les atomes à comparer de la molécule de référence et de la molécule mobile (plusieurs axes de rotation) ainsi que les atomes de la molécule mobile définissant les axes de rotation.

Variable :

Variable interne :

+ TEST : logique; vérifie si les données sont introduites correctement.

Cette procédure est développée dans le fichier ENTREE.FOR .

Procédure **LECDONBIS**

Description :

Cette procédure lit les atomes à comparer de la molécule de référence et de la molécule mobile à égaler (un axe de rotation) ainsi que les atomes de la molécule mobile définissant l'axe de rotation.

Variable :

Variable interne :

+ TEST : logique; vérifie si les données sont introduites correctement.

Cette procédure est développée dans le fichier ENTREE.FOR .

Fonction logique **LIAD** (AXE)

Description :

Cette fonction prend la valeur "vrai" si la liaison autour de laquelle nous effectuons le fitting est admissible c'est-à-dire si la liaison est simple et si elle ne fait partie d'aucun cycle; "fausse" sinon. On vérifie également que l'angle de torsion est bien défini.

Variables :

Paramètre :

+ AXE(4) : entier; désignent les quatre atomes définissant la liaison autour de laquelle nous effectuons le fitting (voir le common LIAISO).

Variables internes :

+ ATYP(9) : entier; contient les types atomiques susceptibles d'engendrer une liaison double ou triple.
+ OK : variable logique qui est mise à "faux" si l'angle de torsion n'est pas défini ou si la liaison autour de laquelle nous effectuons le fitting n'est pas simple ou si elle appartient à un cycle.
+ ANGAD(3) : logique; ANGAD(k) est "faux" si la $k^{ème}$ liaison définissant l'angle de torsion n'existe pas.
+ TYP(2) : logique; TYP(k) est "vrai" si le type du premier (respectivement du second) atome de la liaison autour de laquelle nous effectuons le fitting est l'un des neuf types pouvant engendrer une liaison double ou triple.

+ REP : caractère; si REP='O' ou 'o', alors l'utilisateur confirme que la liaison n'est pas simple.

Cette procédure se trouve dans le fichier TESTAD.FOR .

Procédure **MATROT** (DELTA,CD,ROT)

Description :

Cette procédure calcule la matrice de rotation.

Variables :

Paramètres d'entrée :

+ DELTA : double précision; angle de rotation exprimé en radian.
+ CD(3) : double précision; cosinus directeurs de l'axe de rotation.

Paramètre de sortie :

+ ROT(3,3) : double précision; matrice de rotation.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **MAZDA** (U,V,PILE,N,W)

Description :

Cette procédure met dans PILE tous les triplets (U,V,W) où W appartient à la liste d'adjacence de V et W n'est pas le dernier sommet de cette liste; ce dernier sommet constitue le paramètre de sortie.

Variables :

Paramètres d'entrée :

+ U : entier; sommet précédant le sommet V.
+ V : entier; sommet que nous examinons.
+ PILE(3,ndim2) : entier; pile contenant les triplets (U,V,W).
+ N : entier; pointeur vers la couche supérieure de la pile.

Paramètre de sortie :

+ W : entier; sommet suivant le sommet V. Nous choisissons le dernier sommet appartenant à la liste d'adjacence de V.

Cette procédure est développée dans le fichier TESTAD.FOR .

Procédure **MINENE** (ENEMIN)

Description :

Cette procédure calcule la valeur minimum de l'énergie.

Variable :

Paramètre de sortie :

+ ENEMIN : double précision; minimum de l'énergie.

Cette procédure est développée dans le fichier MINENE.FOR .

Procédure **MINFCT** (RAC)

Description :

Cette procédure calcule le minimum sans contrainte d'énergie de la fonction objectif (somme des carrés des distances entre les atomes à comparer).

Variables :

Paramètre de sortie :

+ RAC : double précision; minimum sans contrainte d'énergie de la fonction objectif; angle exprimé en degré.

Variable interne :

+ RMS : double précision; r.m.s. c'est-à-dire racine carrée de "la valeur optimale de la fonction objectif divisée par le nombre d'atomes à égaler".

Cette procédure est développée dans le fichier MINFCT.FOR .

Procédure **MINFIT1** (ENEMIN,DELTA,DEL,F)

Description :

Cette procédure minimise, sous contrainte d'énergie, la somme des carrés des distances entre les atomes à comparer : l'énergie doit être plus petite ou égale à l'énergie minimum plus une tolérance de 20 kcal/mole.

Variables :

Paramètre d'entrée :

+ ENEMIN : double précision; minimum de l'énergie.

Paramètres de sortie :

+ DELTA(1) : double précision; angle optimum exprimé en radian.
+ DEL : double précision; angle optimum exprimé en degré.
+ F : double précision; valeur optimale de la fonction objectif.

Variables internes :

+ RMS : double précision; r.m.s. c'est-à-dire racine carrée de "la valeur optimale de la fonction objectif divisée par le nombre d'atomes à égaler".

Cette procédure est développée dans le fichier MINFIT1.FOR .

Procédure **MINFIT2** (ZER,RAC,DELTA,DEL,FCT)

Description :

Cette procédure recherche le minimum appartenant à l'intervalle admissible.

Variables :

Paramètres d'entrée :

+ ZER(2) : double précision; bornes du domaine admissible.
+ RAC : double précision; solution de la minimisation sans contrainte d'énergie; cet angle est exprimé en degré.

Paramètres de sortie :

+ DELTA : double précision; solution de la minimisation sous containte d'énergie; cet angle est exprimé en degré.
+ DEL : double précision; solution de la minimisation sous containte d'énergie; cet angle est exprimé en radian.
+ FCT : double précision; valeur optimale de la fonction objectif.

Variables internes :

+ RMS : double précision; r.m.s. c'est-à-dire racine carrée de "la valeur optimale de la fonction objectif divisée par le nombre d'atomes à égaler".
+ F1, F2 : double précision; valeur de la fonction objectif aux bornes du domaine admissible.

Procédure **MINI** (DELTA,DEL)

Description :

Cette procédure calcule le minimum de la somme des carrés des distances entre les atomes à superposer (rotation autour de plusieurs axes).

Variables :

Paramètres de sortie:

+ DELTA(naxe) : double précision; DELTA(i) : angle optimal associé au $i^{ème}$ axe de rotation; ces angles sont exprimés en degré.
+ DEL : double précision; DELTA(i) : angle optimal associé au $i^{ème}$ axe de rotation; ces angles sont exprimés en radian.

Cette procédure est développée dans le fichier MINMUL.FOR .

Procédure **MINMAX** (Z,N)

Description :

Cette procédure retire les éventuels points d'inflexion du tableau des extréma Z et différencie les extréma.

Variables :

Paramètres d'entrée / sortie :

+ Z(maxop,2) : entier; en entrée, la première colonne de Z contient les racines (en degré) de la dérivée première de la fonction d'énergie; en sortie, cette colonne ne contient plus que les extréma de la fonction d'énergie. Le premier élément de cette colonne doit être 0 et le dernier 360. De plus, Z(I,2)=1 si Z(I,1) est un maximum et Z(I,2)=-1 si Z(I,1) est un minimum.
+ N : entier; nombre d'éléments de Z.

Variables internes:

+ L : entier; pointeur vers l'élément de Z que l'on examine.
+ X : double précision; élément de la première colonne de Z exprimé en radian.
+ VAL : double précision; valeur de la dérivée seconde de la fonction d'énergie en X.
+ ERR : double précision; précision exigée.

Cette procédure se trouve dans le fichier MINFIT3.FOR .

Procédure **MOBAT**

Description :

Cette procédure crée ATOMOB, l'ensemble des atomes qui bougent lors de la rotation de la molécule mobile Y autour de chaque axe et complète l'ensemble AME des atomes à comparer en y ajoutant les atomes à comparer se trouvant dans chaque partie mobile de la molécule.

Variables :

Variables internes:

+ DJ : entier; nombre minimal de liaisons séparant les atomes AXE(2,i) et j où j est un atome de la molécule mobile.

+ DK : entier; nombre minimal de liaisons séparant les atomes AXE(3,i) et j où j est un atome de la molécule mobile.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **MOBATBIS**

Description :

Cette procédure crée ATOMOB, l'ensemble des atomes qui bougent lors de la rotation de la molécule mobile Y autour de l'axe défini par les atomes AXE(2) et AXE(4).

Variables :

Variables internes:

+ DJ : entier; nombre minimal de liaisons séparant les atomes AXE(2) et i où i est un atome de la molécule mobile.
+ DK : entier; nombre minimal de liaisons séparant les atomes AXE(3) et i où i est un atome de la molécule mobile.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Fonction logique **MULLIA**

Description :

Cette fonction prend la valeur "faux" si au moins une des liaisons autour desquelles nous effectuons le fitting flexible (définies par AXE) n'est pas admissible (voir la fonction logique LIAD).

Cette fonction se trouve dans le fichier TESTAD.FOR .

Procédure **NODE** (NSO,NAR,BOND,DIST)

Description :

Cette procédure calcule la matrice de distance d'un multigraphe selon la méthode expliquée au paragraphe 5.2.2

Variables :

Paramètres d'entrée :

+ NSO : entier; ordre du multigraphe.
+ NAR : entier; nombre d'arêtes du multigraphe.
+ BOND(2,ndim2) : entier; ensemble des paires de sommets, définissant les arêtes.

Paramètre de sortie :

+ DIST : entier; matrice de distance du multigraphe c'est-à-dire D(i,j)=nombre d'arêtes formant le plus court "chemin" entre les sommets i et j.

Cette procédure est développée dans le fichier NODE.FOR .

Procédure **PALMIER** (NSO,NAR,BOND,X)

Description :

Cette procédure engendre un palmier à partir d'un multigraphe simplement connexe en utilisant une méthode de recherche en profondeur (voir le paragraphe 5.3.2.1.)

Variables :

Paramètres d'entrée:

+ NSO : entier; ordre du multigraphe.
+ NAR : entier; nombre d'arêtes du multigraphe.
+ BOND(2,ndim2) : entier; ensemble des paires de sommets définissant les arêtes.
+ X : entier; sommet à partir duquel nous commençons la recherche en profondeur.

Variables internes :

+ U : entier; sommet précédant le sommet V.
+ V : entier; sommet que nous examinons.
+ W : entier; sommet suivant le sommet V.
+ N : entier; pointeur vers la couche supérieure de la pile PILE (voir la procédure MAZDA).

Cette procédure est développée dans le fichier TESTAD.FOR .

Procédure **PRECH** (X,Y)

Description :

Cette procédure recherche l'ascendant du sommet X dans l'arborescence constituant l'extrémité d'une chaîne passant par X, chaîne dont les arêtes font partie d'un cycle. Le sommet Y est le suivant de cet ascendant, suivant appartenant au chemin reliant l'ascendant à X. L'ascendant est encore noté X (voir le paragraphe 5.4.3.3.).

Cette procédure est développée dans le fichier TESTAD.FOR .

Procédure **PRESEN**

Description :
Cette procédure affiche le titre du programme à l'écran.

Cette procédure est développée dans le fichier PRESEN.FOR .

Procédure **PRODVER** (X,Y,A,B,C)

Description :
Cette procédure calcule le produit vectoriel des vecteurs X et Y.

Variables :

Paramètres d'entrée :
+ X, Y : double précision; coordonnées cartésiennes des vecteurs dont nous calculons le produit vectoriel.

Paramètres de sortie :
+ A, B, C : double précision; composantes du vecteur résultant du produit vectoriel des vecteurs X et Y.

Cette procédure est développée dans le fichier NODE.FOR .

Fonction double précision **RACINE** (THETA,PAS,G)

Description :
Cette fonction prend la valeur de **la** racine de la fonction G située dans l'intervalle]THETA,THETA+PAS] si PAS>0, dans l'intervalle]THETA+PAS,THETA] si PAS<0. La méthode utilisée est celle du balayage (voir paragraphe 3.3.2.1.).

Variables :

Paramètres d'entrée :
+ THETA : double précision; borne de l'intervalle de recherche de la racine.
+ PAS : double précision; "longueur" de l'intervalle (PAS≠0).
+ G : fonction externe double précision dont nous recherchons la racine.

Variables internes :

+ B1,B2 : double précision; bornes de l'intervalle de recherche.
+ G1,G2 : double précision; valeurs de la fonction G aux bornes de l'intervalle de recherche.
+ ERR : double précision; erreur absolue sur la racine.
+ PAS1 : double précision; pas de balayage.

Cette fonction est développée dans le fichier MINFIT3.FOR .

Fonction double précision **RAD** (DELTA)

Description :

Cette fonction convertit en radian un angle exprimé en degré.

Variable :

Paramètre :

+ DELTA : double précision; angle à convertir en radian.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **RECEXT** (Z,N)

Description :

Cette procédure recherche les racines (en degré) de la dérivée première de la fonction d'énergie. Pour plus de renseignements, lire le paragraphe 3.3.2.2.

Variables :

Paramètres de sortie :

+ Z(maxop,2) : entier; la première colonne de Z contient les racines de la dérivée première de la fonction d'énergie. Le premier élément de cette colonne doit être 0 et le dernier 360.
+ N : entier; nombre d'éléments de Z.

Variables internes :

+ DELTA1,DELTA2 : double précision; bornes de l'intervalle que nous examinons.
+ ERR : double précision; erreur absolue sur les racines recherchées.

+ PAS : double précision; longueur des intervalles.
+ G1,G2 : double précision; valeurs de la dérivée aux bornes de l'intervalle.

Cette procédure est développée dans le fichier MINFIT3.FOR .

Procédure **RECOPT** (Z,N,DAUX,DELTA,DEL,SOM)

Description :

Cette procédure recherche le meilleur angle de rotation pour effectuer le fitting avec contrainte d'énergie selon la deuxième approche expliquée au paragraphe 3.3.

Variables :

Paramètres d'entrée :

+ Z(maxop,2) : entier; la première colonne de Z contient les extréma (en degré) de la fonction d'énergie; en outre, le premier élément de cette colonne est 0 et le dernier est 360. De plus, Z(I,2)=1 si Z(I,1) est un maximum et Z(I,2)=-1 si Z(I,1) est un minimum.
+ N : entier; nombre d'éléments de Z.
+ DAUX : double précision; solution du fitting sans contrainte d'énergie.

Paramètres de sortie :

+ DELTA : double précision; solution du fitting avec contrainte d'énergie exprimée en radian.
+ DEL : double précision; solution du fitting avec contrainte d'énergie exprimée en degré.
+ SOM : double précision; valeur de la fonction objectif correspondant à l'angle optimal DELTA trouvé.

Variables internes:

+ SORTIE : entier; prend la valeur 0, 1 ou 2 si respectivement nous nous heurtons à une barrière d'énergie nous empêchant d'atteindre l'intervalle suivant, ou nous nous heurtons à une barrière d'énergie nous empêchant d'atteindre l'angle optimal DAUX, ou nous atteignons l'angle optimal DAUX.
+ A,B : double précision; bornes de l'intervalle que nous examinons.
+ D1,D2 : double précision; solutions (en degré) obtenues en partant de l'angle 0, respectivement de l'angle 360.
+ RMS : double précision; racine carrée de "la valeur optimale de la fonction objectif divisée par le nombre d'atomes à comparer".

Le common UN permet de calculer la fonction E(.)-E(A)-20 (voir la fonction Y).

Cette procédure a été développée dans le fichier MINFIT3.FOR .

Procédure **REPCY** (NSO,NAR,BOND)

Description :

Cette procédure repère les arêtes du multigraphe faisant partie d'un cycle.

Variables :

Paramètres d'entrée:

+ NSO : entier; ordre du multigraphe.
+ NAR : entier; nombre d'arêtes du multigraphe.
+ BOND(2,ndim2) : entier; ensemble des paires de sommets définissant les arêtes.

Variables internes :

+ X1 : entier; extrémité initiale de l'arc que nous examinons.
+ X2 : entier; extrémité terminale de l'arc que nous examinons.
+ I : entier; pointeur vers l'arc courant du tableau PALM.

Cette fonction est développée dans le fichier TESTAD.FOR .

Procédure **RMS** (NORM,FCT,FNORM)

Description :

Cette procédure calcule la distance entre chaque atome à égaler, la somme des carrés de ces distances et le r.m.s.

Variables :

Paramètres de sortie :

+ NORM(ndim1) : double précision; NORM(i) contient la distance séparant le $i^{ème}$ atome à comparer de la molécule fixe et le $i^{ème}$ atome à comparer de la molécule mobile.

+ FCT : double précision; somme des carrés des distances entre les atomes à comparer.
+ FNORM : double précision; r.m.s. c'est-à-dire racine carrée de "la valeur optimale de la somme des carrés des distances entre les atomes à comparer divisée par le nombre d'atomes à comparer.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **RMSBIS** (NORM,FCT,FNORM)

Description :

Cette procédure calcule la distance entre chaque atome à égaler, la somme des carrés de ces distances et le r.m.s.

Variables :

Paramètres de sortie :

+ NORM(ndim1) : double précision; NORM(i) contient la distance séparant le $i^{ème}$ atome à comparer de la molécule fixe et le $i^{ème}$ atome à comparer de la molécule mobile.
+ FCT : double précision; somme des carrés des distances entre les atomes à comparer.
+ FNORM : double précision; r.m.s. c'est-à-dire racine carrée de "la valeur optimale de la somme des carrés des distances entre les atomes à comparer divisée par le nombre d'atomes à comparer.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **ROTATI** (DELTA)

Description :

Cette procédure effectue une rotation de la partie mobile de la molécule mobile autour de chaque axe de rotation.

Variable :

Paramètre d'entrée :

+ DELTA(naxe) : double précision; angles de rotation exprimés en radian.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **ROTATIBIS** (Z,DELTA)

Description :

Cette procédure effectue une rotation de la partie mobile de la molécule mobile.

Variables :

Paramètre d'entrée :

+ DELTA : double précision; angle de rotation exprimé en radian.

Paramètre d'entrée / sortie :

+ Z(ndim1,3) : double précision; molécule mobile.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **TORANG** (Y,AXE,DANG)

Description :

Cette procédure calcule l'angle de torsion formé par le plan déterminé par les atomes Y(AXE(2),.), Y(AXE(3),.), Y(AXE(4),.) et par le plan déterminé par les atomes Y(AXE(1),.), Y(AXE(2),.) et Y(AXE(3),.) (voir annexe).

Variables :

Paramètres d'entrée :

+ Y(ndim1,3) : double précision; contient les coordonnées cartésiennes des atomes de la molécule mobile.
+ AXE(4) : entier; contient les quatre atomes définissant l'angle de torsion.

Paramètre de sortie :

+ DANG : double précision; angle de torsion exprimé en radian.

Variables internes :

+ M(3) : double précision; vecteur normal au plan déterminé par les atomes Y(AXE(2),.), Y(AXE(3),.), Y(AXE(4),.).
+ N(3) : double précision; vecteur normal au plan déterminé par les atomes Y(AXE(1),.), Y(AXE(2),.) et Y(AXE(3),.).
+ D : double précision; norme euclidienne du vecteur $(A,B,C)^t$

Cette procédure est développée dans le fichier NODE.FOR .

Procédure **TRANS** (MAT,NAT,T)

Description :

Cette procédure effectue la translation d'une molécule.

Variables :

Paramètres d'entrée :

+ NAT : entier; nombre d'atomes de la molécule MAT.
+ T(3) : double précision; vecteur de translation.

Paramètre d'entrée / sortie :

+ MAT(ndim1,3) : double précision; molécule à translater.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Procédure **TRANSF** (MAT,AXE,T)

Description :

Cette procédure calcule la translation permettant de faire passer l'axe de rotation par l'origine du système de référence de la molécule mobile.

Variables :

Paramètres d'entrée :

+ AXE(4) : entier; AXE(2) et AXE(3) sont les atomes définissant la liaison simple servant d'axe de rotation.
+ MAT(ndim1) : double précision; molécule mobile.

Paramètre de sortie :

+ T(3) : double précision; vecteur de translation.

Cette procédure est développée dans le fichier GEOMETRIE.FOR .

Fonction logique **VERDEG** (NSO,NAR,BOND)

Description :

Cette fonction prend la valeur "vrai" si les sommets du multigraphe sont de degré au plus quatre. Sinon, elle prend la valeur "faux". Cette fonction crée également, si c'est possible, la structure d'adjacence ADJL du multigraphe (voir le paragraphe 5.3.1.).

Variables :

Paramètres:

+ NSO : entier; ordre du multigraphe.
+ NAR : entier; nombre d'arêtes du multigraphe.
+ BOND(2,ndim2) : entier; ensemble des paires de sommets définissant les arêtes.

Cette fonction est développée dans le fichier TESTAD.FOR .

Fonction logique **VERMS** (NSO)

Description :

Cette fonction prend la valeur "vrai" si le multigraphe est simple; "faux" sinon (voir le paragraphe 5.3.3.)

Variable :

Paramètre :

+ NSO : entier; ordre du multigraphe.

Cette procédure est développée dans le fichier TESTAD.FOR .

Fonction logique **VERSC** (NSO,NAR,BOND)

Description :

Cette fonction prend la valeur "vrai" si le multigraphe est simplement connexe; "faux" sinon (voir le paragraphe 5.3.2.2.).

Variables :

Paramètres:

+ NSO : entier; ordre du multigraphe.
+ NAR : entier; nombre d'arêtes du multigraphe.
+ BOND(2,ndim2) : entier; ensemble des paires de sommets définissant les arêtes.

Cette fonction est développée dans le fichier TESTAD.FOR .

Fonction double précision **Y** (THETA)

Description :

Cette fonction prend la valeur de la fonction E(THETA)-E(A)-20.

Variable :

Paramètre :

+ THETA : double précision; angle exprimé en degré.

Le common UN contenant la variable double précision A est créé dans la procédure RECOPT.

Cette fonction est développée dans le fichier MINFIT3.FOR .

Procédure **ZERO** (ENEMIN,DELTA)

Description :

Cette procédure recherche la racine de la fonction "énergie - ENEMIN - 20" la plus proche de 0 appartenant à l'intervalle $[0,\pi]$ et la racine la plus proche de 0 appartenant à l'intervalle $[-\pi,0]$.

Variables :

Paramètre d'entrée :

+ ENEMIN : double précision; minimum de l'énergie (voir la procédure MINENE).

Paramètre de sortie :

+ DELTA(2) : double précision; DELTA(1) : racine de la fonction "énergie - ENEMIN - 20" la plus proche de 0 appartenant à l'intervalle $[0,\pi]$; DELTA(2) : racine la plus proche de 0 appartenant à l'intervalle $[-\pi,0]$.

Cette procédure est développée dans le fichier ZERO.FOR .

Chapitre IX

Résultats

Nous donnons les résultats de la superposition des deux molécules: méthyl-3 cyclobutène, prise comme molécule de référence et le 1-pentene, prise comme molécule à comparer, en appliquant les quatre méthodes vues précédemment.

Nous avons appliqué une ou plusieurs fois les différents fittings flexibles aux coordonnées des atomes des deux molécules obtenues après avoir effectué un fitting rigide qui a superposé les atomes de carbone 2, 3, 4 sur les atomes de carbone 2', 3', 4'.

Voici une réprésentation de la superposition des deux molécules après ce fitting rigide.

9.1 Application de la première méthode: fitting avec contrainte d'énergie (chap III.2.b)

a) Résultats

---------- MOLECULES A COMPARER ----------

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Fitting sur un axe de rotation avec energie
==========

---------- ITERATION 1 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
5	3.61333	0.29145	0.00000	5	3.61332	-2.61239	0.00000	2.903840

La somme des distances au carre : 8.432287

L'axe de rotation :

3	1.66000	-1.16047	0.00000	4	3.100000	-1.160470	0.000000

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
5	3.61333	0.29145	0.00000	5	3.61332	-0.37138	-1.21877	1.387354

La somme des distances au carre : 1.924751

L'angle de rotation : 122.92089

Le RMS pour les 1 atomes : 1.38735

---------- ITERATION 2 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	-1.21877
6	4.46889	0.53344	-1.25740	2.41555	0.15468	-2.03129
7	4.50926	-0.38004	-1.85073	2.77888	0.71320	-2.89394
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.69266	0.91501
11	4.21889	0.46273	0.88998	3.46332	-2.18680	-0.05237
12	5.47815	0.81890	-0.96074	4.21888	0.46878	-0.87887
13	4.02481	1.33273	-1.85073	4.21888	-1.02536	-1.84624
14				1.81000	0.80866	-1.40383
15				1.81000	-0.68547	-2.37121

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
6	4.46889	0.53344	-1.25740	6	2.41555	0.15468	-2.03129	2.226785

La somme des distances au carre : 4.958571

L'axe de rotation :

4	3.10000	-1.16047	0.00000	5	3.613320	-0.371380	-1.218773

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
6	4.46889	0.53344	-1.25740	6	4.68072	0.64063	-0.76263	0.548783

La somme des distances au carre : 0.301163

L'angle de rotation : 130.90143

Le RMS pour les 1 atomes : 0.54878

---------- ITERATION 3 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	-1.21877
6	4.46889	0.53344	-1.25740	4.68072	0.64063	-0.76263
7	4.50926	-0.38004	-1.85073	5.04405	1.19915	-1.62528
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.69266	0.91501
11	4.21889	0.46273	0.88998	3.46332	-2.18680	-0.05237
12	5.47815	0.81890	-0.96074	4.05145	-1.06128	-1.94002
13	4.02481	1.33273	-1.85073	2.78302	0.16075	-1.68306
14				5.51102	0.10851	-0.29836
15				4.24260	1.33054	-0.04140

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	5.04405	1.19915	-1.62528	1.682459

La somme des distances au carre : 2.830667

L'axe de rotation :

5	3.61332	-0.37138	-1.21877	6	4.680720	0.640632	-0.762626	

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	5.54261	0.58415	-1.42751	1.475334

La somme des distances au carre : 2.176610

L'angle de rotation : 313.21477

Le RMS pour les 1 atomes : 1.47533

---------- RESULTAT FINAL ----------

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	-1.21877
6	4.46889	0.53344	-1.25740	4.68072	0.64063	-0.76263
7	4.50926	-0.38004	-1.85073	5.54261	0.58415	-1.42751
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.69266	0.91501
11	4.21889	0.46273	0.88998	3.46332	-2.18680	-0.05237
12	5.47815	0.81890	-0.96074	4.05145	-1.06128	-1.94002
13	4.02481	1.33273	-1.85073	2.78302	0.16075	-1.68306
14				4.99142	0.40683	0.25565
15				4.26363	1.64721	-0.79317

b) Représentation de la superposition

9.2 Application de la deuxième méthode: fitting pour une énergie admissible au départ (chap III.2.c)

a) Résultats

---------- MOLECULES A COMPARER ----------

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Fitting sur un axe de rotation avec energie
==========

---------- ITERATION 1 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
5	3.61333	0.29145	0.00000	5	3.61332	-2.61239	0.00000	2.903840

La somme des distances au carre : 8.432287

L'axe de rotation :

3	1.66000	-1.16047	0.00000	4	3.100000	-1.160470	0.000000	

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
5	3.61333	0.29145	0.00000	5	3.61332	-0.37138	1.21877	1.387351

La somme des distances au carre : 1.924741

L'angle de rotation : -122.92105

Le RMS pour les 1 atomes : 1.38735

---------- ITERATION 2 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	1.21877
6	4.46889	0.53344	-1.25740	2.41555	0.15469	2.03129
7	4.50926	-0.38004	-1.85073	2.77888	0.71321	2.89393
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-2.18680	0.05237
11	4.21889	0.46273	0.88998	3.46332	-0.69266	-0.91501
12	5.47815	0.81890	-0.96074	4.21888	-1.02535	1.84624
13	4.02481	1.33273	-1.85073	4.21888	0.46878	0.87887
14				1.81000	-0.68547	2.37120
15				1.81000	0.80867	1.40383

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
6	4.46889	0.53344	-1.25740	6	2.41555	0.15469	2.03129	3.895526

La somme des distances au carre : 15.175126

L'axe de rotation :

4	3.10000	-1.16047	0.00000	5	3.613320	-0.371376	1.218771

A T T E N T I O N

L'energie n'est pas admissible en zero

L'execution de cet iteration est arretee

Fitting sur un axe de rotation avec energie

---------- ITERATION 3 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	1.21877
6	4.46889	0.53344	-1.25740	2.41555	0.15469	2.03129
7	4.50926	-0.38004	-1.85073	2.77888	0.71321	2.89393
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-2.18680	0.05237
11	4.21889	0.46273	0.88998	3.46332	-0.69266	-0.91501
12	5.47815	0.81890	-0.96074	4.21888	-1.02535	1.84624
13	4.02481	1.33273	-1.85073	4.21888	0.46878	0.87887
14				1.81000	-0.68547	2.37120
15				1.81000	0.80867	1.40383

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	2.77888	0.71321	2.89393	5.167325

La somme des distances au carre : 26.701253

L'axe de rotation :

5	3.61332	-0.37138	1.21877	6	2.415550	0.154688	2.031288	

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	1.73559	0.69168	1.36992	4.383419

La somme des distances au carre : 19.214365

L'angle de rotation : 127.96426

Le RMS pour les 1 atomes : 4.38342

---------- RESULTAT FINAL ----------

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	1.21877
6	4.46889	0.53344	-1.25740	2.41555	0.15469	2.03129
7	4.50926	-0.38004	-1.85073	1.73559	0.69168	1.36992
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-2.18680	0.05237
11	4.21889	0.46273	0.88998	3.46332	-0.69266	-0.91501
12	5.47815	0.81890	-0.96074	4.21888	-1.02535	1.84624
13	4.02481	1.33273	-1.85073	4.21888	0.46878	0.87887
14				2.77266	0.82853	2.81007
15				1.89061	-0.68380	2.48897

b) Représentation de la superposition

9.3 Application de la troisième méthode: fitting parcourant l'énergie (chap III.3)

a) Résultats

---------- MOLECULES A COMPARER ----------

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Fitting sur un axe de rotation avec energie
==========

---------- ITERATION 1 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
5	3.61333	0.29145	0.00000	5	3.61332	-2.61239	0.00000	2.903840

La somme des distances au carre : 8.432287

L'axe de rotation :

3	1.66000	-1.16047	0.00000	4	3.100000	-1.160470	0.000000	

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
5	3.61333	0.29145	0.00000	5	3.61332	-0.37138	1.21877	1.387353

La somme des distances au carre : 1.924747

L'angle de rotation :

sans energie : 180.00001
avec energie : -122.92096

Le RMS pour les 1 atomes : 1.38735

---------- ITERATION 2 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	1.21877
6	4.46889	0.53344	-1.25740	2.41555	0.15468	2.03129
7	4.50926	-0.38004	-1.85073	2.77888	0.71320	2.89394
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-2.18680	0.05237
11	4.21889	0.46273	0.88998	3.46332	-0.69266	-0.91501
12	5.47815	0.81890	-0.96074	4.21888	-1.02535	1.84624
13	4.02481	1.33273	-1.85073	4.21888	0.46878	0.87887
14				1.81000	-0.68547	2.37120
15				1.81000	0.80867	1.40383

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
6	4.46889	0.53344	-1.25740	6	2.41555	0.15468	2.03129	3.895528

La somme des distances au carre : 15.175142

L'axe de rotation :

4	3.10000	-1.16047	0.00000	5	3.613320	-0.371378	1.218772	

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
6	4.46889	0.53344	-1.25740	6	4.51773	0.77991	0.74110	2.014233

La somme des distances au carre : 4.057135

L'angle de rotation :

sans energie : -122.39058
avec energie : -122.39058

Le RMS pour les 1 atomes : 2.01423

---------- ITERATION 3 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	1.21877
6	4.46889	0.53344	-1.25740	4.51773	0.77991	0.74110
7	4.50926	-0.38004	-1.85073	4.88105	1.33843	1.60375
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-2.18680	0.05237
11	4.21889	0.46273	0.88998	3.46332	-0.69266	-0.91501
12	5.47815	0.81890	-0.96074	2.76638	0.03679	1.77032
13	4.02481	1.33273	-1.85073	4.18345	-1.03589	1.86799
14				3.94759	1.44442	0.09190
15				5.36467	0.37175	0.18957

Les atomes a egaler :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	4.88105	1.33843	1.60375	3.876182

La somme des distances au carre : 15.024789

L'axe de rotation :

5	3.61332	-0.37138	1.21877	6	4.517725	0.779908	0.741100

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	4.74255	0.65008	-0.31754	1.861788

La somme des distances au carre : 3.466256

L'angle de rotation :

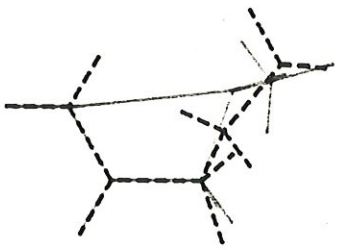
sans energie : 168.82671
avec energie : 168.82671

Le RMS pour les 1 atomes : 1.86179

---------- RESULTAT FINAL ----------

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-0.37138	1.21877
6	4.46889	0.53344	-1.25740	4.51773	0.77991	0.74110
7	4.50926	-0.38004	-1.85073	4.74255	0.65008	-0.31754
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-2.18680	0.05237
11	4.21889	0.46273	0.88998	3.46332	-0.69266	-0.91501
12	5.47815	0.81890	-0.96074	2.76638	0.03679	1.77032
13	4.02481	1.33273	-1.85073	4.18345	-1.03589	1.86799
14				5.44565	0.77423	1.31296
15				4.00512	1.73028	0.88978

b) Représentation de la superposition



9.4 Application de la quatrième méthode: fitting sur plusieurs axes de rotation (chap IV)

a)Résultats

---------- MOLECULES A COMPARER ----------

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Fitting sur plusieurs axes de rotation
==========

---------- ITERATION 1 ----------

*) Donnees de l'iteration :

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	-2.61239	0.00000
6	4.46889	0.53344	-1.25740	2.41555	-3.58034	0.00000
7	4.50926	-0.38004	-1.85073	2.77888	-4.60801	0.00000
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-0.64664	-0.88998
11	4.21889	0.46273	0.88998	3.46332	-0.64664	0.88998
12	5.47815	0.81890	-0.96074	4.21888	-2.78368	-0.88998
13	4.02481	1.33273	-1.85073	4.21888	-2.78368	0.88998
14				1.81000	-3.40906	-0.88998
15				1.81000	-3.40907	0.88998

Les atomes a comparer :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	2.77888	-4.60801	0.00000	4.929011
5	3.61333	0.29145	0.00000	5	3.61332	-2.61239	0.00000	2.903840
6	4.46889	0.53344	-1.25740	6	2.41555	-3.58034	0.00000	4.766597

La somme des distances au carre : 55.447879

Les axes de rotation :

5	3.61332	-2.61239	0.00000	6	2.415550	-3.580340	0.000000
3	1.66000	-1.16047	0.00000	4	3.100000	-1.160470	0.000000
4	3.10000	-1.16047	0.00000	5	3.613320	-2.612390	0.000000

Les 3 angles de rotation initiaux de la minimisation :

60.00000 autour de 5 - 6
60.00000 autour de 3 - 4
60.00000 autour de 4 - 5

*) Les resultats de l'iteration :

NO	Molecule fixe			NO	Molecule mobile			Distance
7	4.50926	-0.38004	-1.85073	7	4.50927	-0.38003	-1.85073	0.000012
5	3.61333	0.29145	0.00000	5	3.61332	0.29145	-0.00001	0.000014
6	4.46889	0.53344	-1.25740	6	4.46889	0.53344	-1.25740	0.000005

La somme des distances au carre : 0.000000

Les 3 angles de rotation :

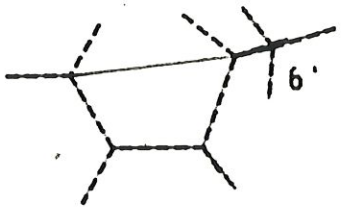
-179.99884	autour de	5 - 6
179.99964	autour de	3 - 4
-120.00073	autour de	4 - 5

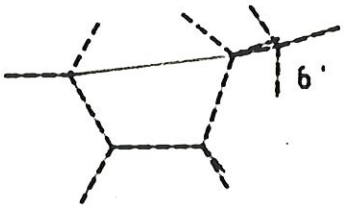
Le RMS pour les 3 atomes : 0.00001

RESULTAT FINAL

NO	Molecule fixe			Molecule mobile		
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.99000	0.00000	0.00000	0.99000	0.00000	0.00000
3	1.66000	-1.16047	0.00000	1.66000	-1.16047	0.00000
4	3.10000	-1.16047	0.00000	3.10000	-1.16047	0.00000
5	3.61333	0.29145	0.00000	3.61332	0.29145	-0.00001
6	4.46889	0.53344	-1.25740	4.46889	0.53344	-1.25740
7	4.50926	-0.38004	-1.85073	4.50927	-0.38003	-1.85073
8	1.16500	-2.01784	0.00000	1.48500	0.85737	0.00000
9	3.46333	-1.67431	0.88998	1.16500	-2.01784	0.00000
10	3.46333	-1.67431	-0.88998	3.46332	-1.67429	0.88998
11	4.21889	0.46273	0.88998	3.46332	-1.67431	-0.88998
12	5.47815	0.81890	-0.96074	2.76554	0.97656	-0.00002
13	4.02481	1.33273	-1.85073	4.21886	0.46275	0.88998
14				5.47814	0.81890	-0.96072
15				4.02482	1.33274	-1.85072

b) Représentation de la superposition





Chapitre X

Programme

Ce chapitre contient le programme reprenant les quatre méthodes développées dans cet ouvrage.

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      PROGRAM FITTIN
      ==============
c
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      COMMON  /FIXE/        XAT,X,XLIEN
      COMMON  /MOBILE/      YAT,Y,YLIEN
      COMMON  /MOLECU/      BOND,NBAT,ITYP,CHGE,TITRE
      COMMON  /DISTA/       DIST
      COMMON  /ADJACENCE/   ADJL
      COMMON  /ENERGI/      EPS,RVDW,QC,VT,ST,NT

      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      INTEGER               XAT,XLIEN,YAT,YLIEN,
     1                      BOND(2,NDIM2),ITYP(NDIM1),NBAT(NDIM1),ST(9),
     2                      DIST(NDIM1,NDIM1),ADJL(0:4,NDIM1)

      DOUBLE PRECISION      X(NDIM1,3),Y(NDIM1,3),CHGE(NDIM1),
     1                      EPS(NDIM1,NDIM1),RVDW(NDIM1),QC(NDIM1),
     2                      VT(9),NT(9)

      CHARACTER*70          TITRE
c
c     Variables internes :
c     ....................
c
      LOGICAL               ENCORE
      INTEGER               ITER,CHOIX
      CHARACTER*70          NOM
c
c
c     Corps :
c     -------
c
c
c...  Presentation du programme.
c
      CALL PRESEN
c
c...  Lecture des deux molecules.
c
      CALL SCLEAR
      WRITE(*,1100)
c
c...  Lecture de la molecule fixe.
c
      CALL DONNEE('fixe',X,XAT,XLIEN)
```

```
c
c...    Lecture de la molecule mobile.
c
        CALL DONNEE('mobile',Y,YAT,YLIEN)

        CALL REPCY(YAT,YLIEN,BOND)
c
c...    Calcul de la matrice de distance.
c
        CALL DISTAN(YAT,YLIEN,BOND,DIST)
c
c...    Lecture du fichier des iterees et impression des molecules initiales.
c
        WRITE(*,1300)
        READ(*,1000) NOM
        OPEN (UNIT=90,FILE=NOM,STATUS='NEW')

        WRITE(90,2100)
        CALL IMPATO
c
c...    Initialisation du nombre d'iteration.
c
        ITER=0
c
c...    Initialisation des parametres d'energie de Van der Waals et de Coulomb.
c
        CALL INIVDW(YAT,YLIEN,NBAT,BOND,RVDW,EPS)
        CALL INICOUL(YAT,NBAT,CHGE,QC)
c
c...    La variable ENCORE est vraie si l'utilisateur desire effectuer un autre
c...    fitting flexible.
c
        ENCORE = .TRUE.

        DO 100 WHILE (ENCORE)
c
c...    Choix du fitting.
c
          CHOIX=0
          DO 10 WHILE (CHOIX.LT.1.OR.CHOIX.GT.4)
             CALL SCLEAR
             WRITE(*,1200)
             READ(*,*) CHOIX
10        CONTINUE
c
c...      Appel aux routines correspondant  au choix de l'utilisateur.
c
          IF (CHOIX.EQ.1) THEN
             CALL FITMUL(ITER)
          ELSEIF (CHOIX.EQ.2) THEN
             CALL FITENE1(ITER)
          ELSEIF (CHOIX.EQ.3) THEN
             CALL FITENE2(ITER)
          ELSE
             CALL FITENE3(ITER)
          ENDIF
```

```
c
c...      L'utilisateur choisit s'il veut effectuer un nouveau fitting.
c
          WRITE(*,1500)
          READ (*,1000) NOM
          ENCORE=NOM.EQ.'OUI'.OR.NOM.EQ.'oui'

100       CONTINUE
c
c...      Impression sur le fichier des iterees des deux molecules, resultats
c...      des fitting flexibles.
c
          WRITE(90,3200)
          CALL IMPATO
c
c...      Ecriture de la nouvelle molecule mobile sur fichier.
c
          CALL SCLEAR
          WRITE(*,1550)
          CALL ECRITU

          STOP
c
c         Formats :
c         ---------
c
1000      FORMAT(A)
1100      FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'!',T54,'!',/,1X,T25,
          1       '!',T31,'Entree des donnees',T54,'!',/,1X,T25,'!',T54,'!',
          2       /,1X,T25,'+',28('-'),'+')
1200      FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'!',T54,'!',/,1X,T25,
          1       '!',T32,'Choix du fitting',T54,'!',/,1X,T25,'!',T54,'!',
          2       /,1X,T25,'+',28('-'),'+',//,1X,T5,'Vous pouvez choisir,',
          3       'un fitting :',/,1X,T20,'1. sur plusieurs axes de ',
          4       'rotation,',/,1X,T20,'2. avec contraintes d''energie,',/,
          5       1X,T20,'3. pour une energie admissible en zero,',/,1X,T20,
          6       '4. en parcourant l''energie.',//,1X,T5,'Votre choix ',
          7       '(1..4) : ',$)
1300      FORMAT(/,1X,T5,'Introduisez le nom du fichier sur lequel seront'
          1       ,' incrites les iterees.',/,1X,T5,'NOM : ',$)
1500      FORMAT(//,1X,T5,'Desirez-vous effectuer un AUTRE fitting sur ',
          1       'ces deux molecules ?',/,1X,T5,'Veuillez repondre par  ',
          2       '"OUI"  ou  "NON" : ',$)
1550      FORMAT(1X,T25,'+',27('-'),'+',/,1X,T25,'!',T53,'!',/,1X,T25,
          1       '!',T31,'Fin du programmme',T53,'!',/,1X,T25,'!',T53,'!',
          2       /,1X,T25,'+',27('-'),'+')

2100      FORMAT(1X,26('-'),4X,'MOLECULES A COMPARER',4X,25('-'),///)
3200      FORMAT(///,1X,29('-'),4X,'RESULTAT FINAL',4X,28('-'),///)

          END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE FITENE1 (ITER)
c     =========================
c
c     Description :
c     -------------
c
c     Ce programme effectue un fitting flexible sous contrainte d'energie.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      INTEGER           ITER
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      COMMON  /FIXE/     XAT,X,XLIEN
      COMMON  /MOBILE/   YAT,Y,YLIEN
      COMMON  /LIAISO/   AXE,ATOMOB
      COMMON  /DISTA/    DIST
      COMMON  /ENERGI/   EPS,RVDW,QC,VT,ST,NT
      COMMON  /DIVISI/   NBATEG,AME,AFE
      COMMON  /MOLECU/   BOND,NBAT,ITYP,CHGE,TITRE
      COMMON  /TRANF/    CD,T

      INTEGER            XAT,XLIEN,YAT,YLIEN,
     1                   BOND(2,NDIM2),NBAT(NDIM1),ITYP(NDIM1),
     2                   ATOMOB(0:NDIM1),NBATEG,AME(NDIM1),AFE(NDIM1),
     3                   AXE(4),DIST(NDIM1,NDIM1),ST(9)
      DOUBLE PRECISION   X(NDIM1,3),Y(NDIM1,3),CHGE(NDIM1),CD(3),T(3),
     1                   EPS(NDIM1,NDIM1),RVDW(NDIM1),QC(NDIM1),
     2                   VT(9),NT(9)
      CHARACTER*70       TITRE
c
c     Variables internes :
c     ....................
c
      INTEGER            I,J
      DOUBLE PRECISION   ENEMIN,DELTA,DEL,FCT,FNORM
      LOGICAL            ENCORE
      CHARACTER*70       NOM
```

```
c
c
c       Corps :
c       -------
c
        WRITE(90,2000)
c
c...    La variable ENCORE est vraie si l'utilisateur desire effectuer un
c...    nouveau fitting flexible.
c
        ENCORE = .TRUE.

        DO 100 WHILE (ENCORE)

           ITER=ITER+1
c
c...       Impression a l'ecran du nombre d'iterations.
c
           CALL SCLEAR
           WRITE(*,1350) ITER
c
c...       Entree de l'axe de rotation et des atomes a egaler.
c
           CALL LECDONBIS

c
c...       Impression du nombre d'iterations sur le fichier des iterees,
c...                 des coordonnees des atomes des deux molecules,
c...                 des atomes a comparer,
c...                 de l'axe de rotation.
c
           WRITE(90,2300) ITER
           CALL IMPATO
           WRITE(90,2400)
           CALL IMPEGABIS(.FALSE.,FNORM)
           WRITE(90,2600)
           WRITE(90,2700) AXE(2),(Y(AXE(2),J),J=1,3),AXE(3),
     1                    (Y(AXE(3),J),J=1,3)
c
c...       Initialisation des parametres d'energie de Torsion: VT, NT, ST.
c
           CALL INITOR(AXE,ITYP,VT,NT,ST)

c
c...       Determination de la partie mobile de la molecule mobile.
c
           CALL MOBATBIS

c
c...       Calcul des cosinus directeurs.
c
           CALL COSDIR (Y,AXE,CD)

c
c...       Translation des coordonnees des atomes des deux molecules.
c
           T(1)=Y(AXE(3),1)
           T(2)=Y(AXE(3),2)
           T(3)=Y(AXE(3),3)
           CALL TRANS(X,XAT,T)
           CALL TRANS(Y,YAT,T)
```

```
c
c          Calcul de la minimisation.
c          --------------------------
c
           CALL SCLEAR
           WRITE(*,1400)
c
c          1. Calcul de la valeur minimale ENEMIN de l'energie.
c
           CALL MINENE(ENEMIN)
c
c          2. Calcul de l'angle optimal (DELTA en radian, DEL en degre) et
c             de la valeur de la fonction objectif a l'optimum FCT.
c
           CALL MINFIT1(ENEMIN,DELTA,DEL,FCT)
c
c...       Mise a jour de la molecule fixe.
c
           DO 10 I=1,3
              T(I)=-T(I)
10         CONTINUE

           CALL TRANS(X,XAT,T)
c
c...       Mise a jour de la molecule mobile.
c
           CALL ROTATIBIS(Y,DELTA)
           CALL TRANS(Y,YAT,T)
c
c          Sortie des resultats obtenus.
c          -----------------------------
c
c...       Impression des atomes a comparer dans le fichier de iterees,
c...                 de l'angle optimal,
c...                 du rms.
c
           WRITE(90,2800)
           CALL IMPEGABIS(.TRUE.,FNORM)
           WRITE(90,2900) DEL
           WRITE(90,3100) NBATEG,FNORM
c
c...       L'utilisateur choisit s'il veut effectuer un nouveau fitting.
c
           CALL SCLEAR
           WRITE(*,1450)
           WRITE(*,1500)
           READ(*,1000) NOM
           ENCORE=NOM.EQ.'OUI'.OR.NOM.EQ.'oui'

100     CONTINUE

        RETURN
```

```
c
c       Formats :
c       ---------
c
1000    FORMAT(A)
1350    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T34,'Iteration ',I2,T54,'¦',/,1X,T25,'¦',
        2       T54,'¦',/,1X,T25,'+',28('-'),'+')
1400    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T34,'Minimisation',T54,'¦',/,1X,T25,'¦',T54,'¦',
        2       /,1X,T25,'+',28('-'),'+')
1450    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T33,'Fin d''iteration',T54,'¦',/,1X,T25,'¦',T54,'¦',
        2       /,1X,T25,'+',28('-'),'+')
1500    FORMAT(//,1X,T5,'Desirez-vous effectuer un nouveau fitting sur ',
        1       'ces deux molecules ?',/,1X,T5,'Veuillez repondre par  ',
        2       '"OUI"  ou  "NON" : ',$)

2000    FORMAT(//1X,T19,'Fitting sur un axe de rotation avec energie',/,1X,
        1       T18,45('='))
2300    FORMAT(///,1X,30('-'),4X,'ITERATION ',I2,4X,29('-'),///,1X,
        1       '*) Donnees de l''iteration :',/,1X,T4,26('~'),/)
2400    FORMAT(//,1X,T5,'Les atomes a egaler :',/,1X,T4,23('-'))
2600    FORMAT(//,1x,T5,'L''axe de rotation :',/,1X,T4,21('-'))
2700    FORMAT(1X,T5,I2,3X,3F10.5,T44,I2,3X,3F10.6)
2800    FORMAT(///,1X,'*) Les resultats de l''iteration :'
        1       ,/,1X,T4,32('~'),/)
2900    FORMAT(/,1X,T5,'L''angle de rotation :',T30,F10.5,
        1       /,1X,T4,23('-'))
3100    FORMAT(/,1X,T5,'Le RMS pour les ',I2,' atomes : ',F10.5
        1       ,/,1X,T4,29('-'))

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
c
      SUBROUTINE FITENE2 (ITER)
c     =========================
c
c     Description :
c     -------------
c
c     Ce programme effectue un fitting flexible sous contrainte d'energie.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      INTEGER           ITER
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      COMMON  /FIXE/     XAT,X,XLIEN
      COMMON  /MOBILE/   YAT,Y,YLIEN
      COMMON  /LIAISO/   AXE,ATOMOB
      COMMON  /DISTA/    DIST
      COMMON  /ENERGI/   EPS,RVDW,QC,VT,ST,NT
      COMMON  /DIVISI/   NBATEG,AME,AFE
      COMMON  /MOLECU/   BOND,NBAT,ITYP,CHGE,TITRE
      COMMON  /TRANF/    CD,T

      INTEGER            XAT,XLIEN,YAT,YLIEN,
      1                  BOND(2,NDIM2),NBAT(NDIM1),ITYP(NDIM1),
      2                  ATOMOB(0:NDIM1),NBATEG,AME(NDIM1),AFE(NDIM1),
      3                  AXE(4),DIST(NDIM1,NDIM1),ST(9)
      DOUBLE PRECISION   X(NDIM1,3),Y(NDIM1,3),CHGE(NDIM1),CD(3),T(3),
      1                  EPS(NDIM1,NDIM1),RVDW(NDIM1),QC(NDIM1),
      2                  VT(9),NT(9)
      CHARACTER*70       TITRE
c
c     Variables internes :
c     ....................
c
      INTEGER            I,J
      DOUBLE PRECISION   ENEMIN,DELTA,DEL,RAC,ZER(2),FCT,FNORM,
      1                  ETOR,EVDW,ECOUL
      LOGICAL            ENCORE
      CHARACTER*70       NOM
```

```
c
c
c       Corps :
c       -------
c
        WRITE(90,2000)
c
c...    La variable ENCORE est vraie si l'utilisateur desire effectuer un
c...    nouveau fitting flexible.
c
        ENCORE = .TRUE.

        DO 100 WHILE (ENCORE)

           ITER=ITER+1
c
c...       Impression a l'ecran du nombre d'iterations.
c
           CALL SCLEAR
           WRITE(*,1350) ITER
c
c...       Entree de l'axe de rotation et des atomes a egaler.
c
           CALL LECDONBIS
c
c...       Impression du nombre d'iterations sur le fichier des iterees,
c...                 des coordonnees des atomes des deux molecules,
c...                 des atomes a comparer,
c...                 de l'axe de rotation.
c
           WRITE(90,2300) ITER
           CALL IMPATO
           WRITE(90,2400)
           CALL IMPEGABIS(.FALSE.,FNORM)
           WRITE(90,2600)
           WRITE(90,2700) AXE(2),(Y(AXE(2),J),J=1,3),AXE(3),
     1                    (Y(AXE(3),J),J=1,3)
c
c...       Initialisation des parametres d'energie de Torsion: VT, NT, ST.
c
           CALL INITOR(AXE,ITYP,VT,NT,ST)
c
c...       Determination de la partie mobile de la molecule mobile.
c
           CALL MOBATBIS
c
c...       Calcul des cosinus directeurs.
c
           CALL COSDIR (Y,AXE,CD)
c
c...       Translation des coordonnees des atomes des deux molecules.
c
           T(1)=Y(AXE(3),1)
           T(2)=Y(AXE(3),2)
           T(3)=Y(AXE(3),3)
           CALL TRANS(X,XAT,T)
           CALL TRANS(Y,YAT,T)
```

```
c
c          Calcul de la minimisation.
c          --------------------------
c
           CALL SCLEAR
           WRITE(*,1400)
c
c          1. Calcul de la valeur minimum ENEMIN de l'energie.
c
           CALL MINENE(ENEMIN)
c
c...       Verification de l'energie de conformation en zero.
c
           CALL CONT (0.DO,ETOR,EVDW,ECOUL)
           IF ((ETOR+EVDW+ECOUL).GT.(20+ENEMIN)) THEN
c
c...          Impression d'un message.
c
              WRITE(*,1100)
              WRITE(90,*)
              WRITE(90,1100)
              CALL WAIT(10.0)
c
c...          Mise a jour des molecules fixe et mobile.
c
              DO 10 I=1,3
                 T(I)=-T(I)
10            CONTINUE

              CALL TRANS(X,XAT,T)
              CALL TRANS(Y,YAT,T)
              GOTO 200
           ENDIF
c
c          2. Calcul de l'intervalle admissible [ZER(1),ZER(2)]
c
           CALL ZERO(ENEMIN,ZER)
c
c          3. Calcul de la solution RAC (degre) sans contrainte d'energie.
c
           CALL MINFCT(RAC)
c
c          4. Calcul de l'angle optimal (DELTA en radian, DEL en degre) et
c             de la valeur de la fonction objectif a l'optimum FCT.
c
           CALL MINFIT2(ZER,RAC,DELTA,DEL,FCT)

c
c...       Mise a jour de la molecule fixe.
c
           DO 5 I=1,3
              T(I)=-T(I)
5          CONTINUE

           CALL TRANS(X,XAT,T)
```

```
c
c...        Mise a jour de la molecule mobile.
c
            CALL ROTATIBIS(Y,DELTA)
            CALL TRANS(Y,YAT,T)
c
c           Sortie des resultats obtenus.
c           -----------------------------
c
c...        Impression des atomes a comparer dans le fichier de iterees,
c...                    de l'angle optimal,
c...                    du rms.
c
            WRITE(90,2800)
            CALL IMPEGABIS(.TRUE.,FNORM)
            WRITE(90,2900) DEL
            WRITE(90,3100) NBATEG,FNORM
c
c...        L'utilisateur choisit s'il veut effectuer un nouveau fitting.
c
            CALL SCLEAR
            WRITE(*,1450)
            WRITE(*,1500)
            READ(*,1000) NOM
            ENCORE=NOM.EQ.'OUI'.OR.NOM.EQ.'oui'

100         CONTINUE

200         CONTINUE


            RETURN
```

```
c
c       Formats :
c       ---------
c
1000    FORMAT(A)
1100    FORMAT(1X,79('.'),/,1X,'.',T30,'A T T E N T I O N',T80,'.',
        1       /,1X,'.',T20,'L''energie n''est pas admissible en zero',
        2       T80,'.',/,1X,'.',T20,'L''execution de cet iteration est',
        3       ' arretee',T80,'.',/,1X,79('.'),/)
1350    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T34,'Iteration ',I2,T54,'¦',/,1X,T25,'¦',
        2       T54,'¦',/,1X,T25,'+',28('-'),'+')
1400    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T34,'Minimisation',T54,'¦',/,1X,T25,'¦',T54,'¦',
        2       /,1X,T25,'+',28('-'),'+')
1450    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T33,'Fin d''iteration',T54,'¦',/,1X,T25,'¦',T54,'¦',
        2       /,1X,T25,'+',28('-'),'+')
1500    FORMAT(//,1X,T5,'Desirez-vous effectuer un nouveau fitting sur ',
        1       'ces deux molecules ?',/,1X,T5,'Veuillez repondre par  ',
        2       '"OUI"  ou  "NON" : ',$)

2000    FORMAT(//1X,T19,'Fitting sur un axe de rotation avec energie',/,1X,
        1       T18,45('='))
2300    FORMAT(///,1X,30('-'),4X,'ITERATION ',I2,4X,29('-'),///,1X,
        1       '*) Donnees de l''iteration :',/,1X,T4,26('~'),/)
2400    FORMAT(//,1X,T5,'Les atomes a egaler :',/,1X,T4,23('-'))
2600    FORMAT(//,1x,T5,'L''axe de rotation :',/,1X,T4,21('-'))
2700    FORMAT(1X,T5,I2,3X,3F10.5,T44,I2,3X,3F10.6)
2800    FORMAT(///,1X,'*) Les resultats de l''iteration :'
        1       ,/,1X,T4,32('~'),/)
2900    FORMAT(/,1X,T5,'L''angle de rotation :',T30,F10.5,
        1       /,1X,T4,23('-'))
3100    FORMAT(/,1X,T5,'Le RMS pour les ',I2,' atomes : ',F10.5
        1       ,/,1X,T4,29('-'))

        END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE FITENE3 (ITER)
      =========================
c
c
c     Description :
c     -------------
c
c     Ce programme effectue un fitting flexible en recherchant une solution
c     admissible du point de vue energie.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      INTEGER           ITER
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      COMMON  /FIXE/    XAT,X,XLIEN
      COMMON  /MOBILE/  YAT,Y,YLIEN
      COMMON  /MOLECU/  BOND,NBAT,ITYP,CHGE,TITRE
      COMMON  /DISTA/   DIST
      COMMON  /LIAISO/  AXE,ATOMOB
      COMMON  /ENERGI/  EPS,RVDW,QC,VT,ST,NT
      COMMON  /DIVISI/  NBATEG,AME,AFE
      COMMON  /TRANF/   CD,T

      INTEGER           XAT,XLIEN,YAT,YLIEN,
     1                  BOND(2,NDIM2),NBAT(NDIM1),ITYP(NDIM1),
     2                  ATOMOB(0:NDIM1),NBATEG,AME(NDIM1),AFE(NDIM1),
     3                  AXE(4),DIST(NDIM1,NDIM1),ST(9)
      DOUBLE PRECISION  X(NDIM1,3),Y(NDIM1,3),CHGE(NDIM1),CD(3),T(3),
     1                  EPS(NDIM1,NDIM1),RVDW(NDIM1),QC(NDIM1),
     2                  VT(9),NT(9)
      CHARACTER*70      TITRE
c
c     Variables internes :
c     ....................
c
      PARAMETER  MAXOP = 40

      INTEGER           N,I,J
      DOUBLE PRECISION  DELTA,DEL,RAC,ZER(2),RESENE,RES,Z(MAXOP,2),
     1                  DEG,RAD,FNORM
      LOGICAL           ENCORE,TEST,LIAD,VERDEG,VERMS,VERSC
      CHARACTER*70      NOM
```

```
c
c
c       Corps :
c       -------
c
        WRITE(90,2000)
c
c...    La variable ENCORE est vraie si l'utilisateur desire effectuer un
c...    nouveau fitting flexible.
c
        ENCORE = .TRUE.

        DO 100 WHILE (ENCORE)

           ITER=ITER+1
c
c...       Impression a l'ecran du nombre d'iterations.
c
           CALL SCLEAR
           WRITE(*,1350) ITER
c
c...       Entree de l'axe de rotation et des atomes a egaler.
c
           CALL LECDONBIS
c
c...       Impression du nombre d'iterations sur le fichier des iterees,
c...                 des coordonnees des atomes des deux molecules,
c...                 des atomes a comparer,
c...                 de l'axe de rotation.
c
           WRITE(90,2300) ITER
           CALL IMPATO
           WRITE(90,2400)
           CALL IMPEGABIS(.FALSE.,FNORM)
           WRITE(90,2600)
           WRITE(90,2700) AXE(2),(Y(AXE(2),J),J=1,3),AXE(3),
     1                    (Y(AXE(3),J),J=1,3)
c
c...       Initialisation des parametres d'energie de Torsion: VT, NT, ST.
c
           CALL INITOR(AXE,ITYP,VT,NT,ST)
c
c...       Determination de la partie mobile de la molecule mobile.
c
           CALL MOBATBIS
c
c...       Calcul des cosinus directeurs.
c
           CALL COSDIR (Y,AXE,CD)
c
c...       Translation des coordonnees des atomes des deux molecules.
c
           T(1)=Y(AXE(3),1)
           T(2)=Y(AXE(3),2)
           T(3)=Y(AXE(3),3)
           CALL TRANS(X,XAT,T)
           CALL TRANS(Y,YAT,T)
```

```
c
c          Calcul de la minimisation.
c          --------------------------
c
c
           CALL SCLEAR
           WRITE(*,1400)
c
c...       1. Calcul du minimum de la distance sans contraintes d'energie.
c
           CALL MINFCT(RAC)
c
c...       2. Recherche du minimum de la distance sous contraintes d'energie.
c
c...       2.1. Recherche des extrema  de l'energie sur [0,2pi].
c
           CALL RECEXT(Z,N)
c
c...       2.2. Caracterisation des extrema et elimination des eventuels points
c...            d'inflexion.
c
           CALL MINMAX(Z,N)
c
c...       2.3. Recherche du minimum DELTA radian), DEL (degre) de la distance
c...            sous contrainte d'energie.
c
           CALL RECOPT(Z,N,RAC,DELTA,DEL,RESENE)
c
c...       Calcul de la translation inverse.
c
           DO 10 I=1,3
              T(I)=-T(I)
10         CONTINUE
c
c...       Translation de la molecule fixe.
c
           CALL TRANS(X,XAT,T)
c
c...       Nouvelle conformation de la molecule mobile.
c
           CALL ROTATIBIS(Y,DELTA)
           CALL TRANS(Y,YAT,T)
c
c          Sortie des resultats.
c          ---------------------
c
c...       Impression des atomes a comparer dans le fichier de iterees,
c...                 de l'angle optimal,
c...                 du rms.
c
           WRITE(90,2800)
           CALL IMPEGABIS(.TRUE.,FNORM)
           WRITE(90,2900)
           WRITE(90,2950) RAC
           WRITE(90,3000) DEL
           WRITE(90,3100) NBATEG,FNORM
```

```
c
c...        L'utilisateur choisit s'il veut effectuer un nouveau fitting.
c
            CALL SCLEAR
            WRITE(*,1450)
            WRITE(*,1500)
            READ(*,1000) NOM
            ENCORE=NOM.EQ.'OUI'.OR.NOM.EQ.'oui'

100         CONTINUE

            RETURN
c
c           Formats :
c           ---------
c
1000        FORMAT(A)
1350        FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'!',T54,'!',/,1X,T25,
           1       '!',T34,'Iteration ',I2,T54,'!',/,1X,T25,'!',
           2       T54,'!',/,1X,T25,'+',28('-'),'+')
1400        FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'!',T54,'!',/,1X,T25,
           1       '!',T34,'Minimisation',T54,'!',/,1X,T25,'!',T54,'!',
           2       /,1X,T25,'+',28('-'),'+')
1450        FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'!',T54,'!',/,1X,T25,
           1       '!',T33,'Fin d''iteration',T54,'!',/,1X,T25,'!',T54,'!',
           2       /,1X,T25,'+',28('-'),'+')
1500        FORMAT(//,1X,T5,'Desirez-vous effectuer un nouveau fitting sur ',
           1       'ces deux molecules ?',/,1X,T5,'Veuillez repondre par  ',
           2       '"OUI"  ou  "NON" : ',$)

2000        FORMAT(//1X,T19,'Fitting sur un axe de rotation avec energie',/,1X,
           1       T18,45('='))
2300        FORMAT(///,1X,30('-'),4X,'ITERATION ',I2,4X,29('-'),///,1X,
           1       '*) Donnees de l''iteration :',/,1X,T4,26('~'),/)
2400        FORMAT(//,1X,T5,'Les atomes a egaler :',/,1X,T4,23('-'))
2600        FORMAT(//,1x,T5,'L''axe de rotation :',/,1X,T4,21('-'))
2700        FORMAT(1X,T5,I2,3X,3F10.5,T44,I2,3X,3F10.6)
2800        FORMAT(///,1X,'*) Les resultats de l''iteration :'
           1       ,/,1X,T4,32('~'),/)
2900        FORMAT(/,1X,T5,'L''angle de rotation :',/,1X,T4,23('-'))
2950        FORMAT(1X,T10,'sans energie : ',F10.5)
3000        FORMAT(1X,T10,'avec energie : ',F10.5)
3100        FORMAT(/,1X,T5,'Le RMS pour les ',I2,' atomes : ',F10.5
           1       ,/,1X,T4,29('-'))

            END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE FITMUL (ITER)
c     ================
c
c     Description :
c     -------------
c
c     Cette routine effectue un fitting flexible en considerant plusieurs
c     liaisons simples comme axe de rotation.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      INTEGER           ITER
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200
      PARAMETER  NAXE  = 10

      COMMON  /FIXE/    XAT,X,XLIEN
      COMMON  /MOBILE/  YAT,Y,YLIEN
      COMMON  /MOLECU/  BOND,NBAT,ITYP,CHGE,TITRE
      COMMON  /LIAISO/  NBRAXE,AXE,ATOMOB
      COMMON  /DISTA/   DIST
      COMMON  /DIVISI/  NBATEG,AME,AFE

      INTEGER           XAT,XLIEN,YAT,YLIEN,
     1                  BOND(2,NDIM2),NBAT(NDIM1),ITYP(NDIM1),
     2                  NBATEG,AME(0:NDIM1,0:NAXE),AFE(NDIM1),
     3                  NBRAXE,AXE(4,NAXE),ATOMOB(0:NDIM1,NAXE),
     4                  DIST(NDIM1,NDIM1)
      DOUBLE PRECISION  X(NDIM1,3),Y(NDIM1,3),CHGE(NDIM1)
      CHARACTER*70      TITRE
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
      DOUBLE PRECISION  DELTA(NAXE),FNORM,NORM(NDIM1),FCT,DEL(NAXE)
      LOGICAL           ENCORE
      CHARACTER*70      NOM
```

```
c
c
c       Corps :
c       -------
c
        WRITE(90,2000)
c
c...    La variable ENCORE est vraie si l'utilisateur desire effectuer un
c...    nouveau fitting flexible.
c
        ENCORE=.TRUE.

        DO 40 WHILE (ENCORE)

          ITER=ITER+1
c
c...      Impression du nombre d'iterations a l'ecran.
c
          CALL SCLEAR
          WRITE(*,1350) ITER
c
c...      Entree des axes de rotation et des atomes a egaler.
c
          CALL LECDON
c
c...      Impression du nombre d'iterations sur le fichier des iterees,
c...                des coordonnees des atomes des deux molecules,
c...                des atomes a comparer,
c...                des axes de rotation.
c
          WRITE(90,2300) ITER
          CALL IMPATO
          WRITE(90,2400)
          CALL IMPEGA(.FALSE.,FNORM)
          WRITE(90,2600)
          WRITE(90,2700) (AXE(2,I),(Y(AXE(2,I),J),J=1,3),AXE(3,I),
     1                    (Y(AXE(3,I),J),J=1,3),I=1,NBRAXE)
c
c...      Determination de la partie mobile de la molecule mobile.
c
          CALL MOBAT
```

```
c
c          Calcul de la minimisation.
c          --------------------------
c
           CALL SCLEAR
           WRITE(*,1400)
c
c...       Cette procedure calcule la minimisation de la somme des distances
c...       au carre entre les atomes a comparer.
c
           CALL MINI(DELTA,DEL)
c
c...       Nouvelle conformation de la molecule mobile.
c
           CALL ROTATI(DELTA)
c
c          Sortie des resultats.
c          ---------------------
c
c...       Impression des atomes a comparer dans le fichier des iterees,
c...                 des angles optimaux,
c...                 du rms.
c
           WRITE(90,2800)
           CALL IMPEGA(.TRUE.,FNORM)
           WRITE(90,2900) NBRAXE
           WRITE(90,3000) (DEL(I),AXE(2,I),AXE(3,I),I=1,NBRAXE)
           WRITE(90,3100) NBATEG,FNORM
c
c...       L'utilisateur choisit s'il veut effectuer un nouveau fitting.
c
           CALL SCLEAR
           WRITE(*,1450)
           WRITE(*,1500)
           READ(*,1000) NOM
           ENCORE=NOM.EQ.'OUI'.OR.NOM.EQ.'oui'

40         CONTINUE


           RETURN
```

```
c
c       Formats :
c       ---------
c
1000    FORMAT(A)
1350    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T34,'Iteration ',I2,T54,'¦',/,1X,T25,'¦',
        2       T54,'¦',/,1X,T25,'+',28('-'),'+')
1400    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T34,'Minimisation',T54,'¦',/,1X,T25,'¦',T54,'¦',
        2       /,1X,T25,'+',28('-'),'+')
1450    FORMAT(1X,T25,'+',28('-'),'+',/,1X,T25,'¦',T54,'¦',/,1X,T25,
        1       '¦',T33,'Fin d''iteration',T54,'¦',/,1X,T25,'¦',T54,'¦',
        2       /,1X,T25,'+',28('-'),'+')
1500    FORMAT(//,1X,T5,'Desirez-vous effectuer un nouveau fitting sur ',
        1       'ces deux molecules ?',/,1X,T5,'Veuillez repondre par  ',
        2       '"OUI"  ou  "NON" : ',$)

2000    FORMAT(//,1X,T22,'Fitting sur plusieurs axes de rotation',/,1X,
        1       T21,40('='))
2300    FORMAT(///,1X,30('-'),4X,'ITERATION ',I2,4X,29('-'),///,1X,
        1       '*) Donnees de l''iteration :',/,1X,T4,26('~'),/)
2400    FORMAT(//,1X,T5,'Les atomes a comparer :',/,1X,T4,25('-'))
2600    FORMAT(//,1x,T5,'Les axes de rotation :',/,1X,T4,24('-'))
2700    FORMAT(1X,T5,I2,3X,3F10.5,T44,I2,3X,3F10.6)
2800    FORMAT(///,1X,'*) Les resultats de l''iteration :'
        1       ,/,1X,T4,32('~'),/)
2900    FORMAT(/,1X,T5,'Les ',I2,' angles de rotation :',/,1X,T4,29('-'))
3000    FORMAT(1X,T20,F10.5,'  autour de  ',I2,' - ',I2)
3100    FORMAT(/,1X,T5,'Le RMS pour les ',I2,' atomes : ',F10.5
        1       ,/,1X,T4,29('-'))

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE MINENE (ENEMIN)
c     ==========
c
c     Description :
c     -------------
c
c     Cette procedure calcule la valeur (ENEMIN) minimun de l'energie.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      DOUBLE PRECISION  ENEMIN
c
c     Variables internes :
c     ....................
c
      INTEGER           N,M,MEQ,LCN,MAXFUN,IPRINT,INF,LW
      DOUBLE PRECISION  ETOR,ETOR1,EVDW,EVDW1,ECOUL,ECOUL1,
     1                  DELTA(1),F,G(1),C(2),CN(5,2),ACC,W(55),PI,
     2                  DEG,RAD
c
c
c     Corps :
c     -------
c
c     1. Assignation des parametres de la procedure VF02AD.
c
      PI=ACOS(-1.0)
      LW=55
      N=1
      M=2
      MEQ=0
      INF=-1
      LCN=5
      MAXFUN=100
      ACC=1.D-03
      IPRINT=-1
c
c...  Initialisation de l'angle optimal.
c
      WRITE(*,1300)
      WRITE(*,1400)
      READ (*,1200) DELTA(1)
      DELTA(1)=RAD(DELTA(1))
```

```
c
c       2. Recherche du minimum.
c
        DO 10 WHILE (INF.LE.0)
c
c...       Calcul de la valeur de l'energie au point DELTA(1).
c
           CALL CONT(DELTA(1),ETOR,EVDW,ECOUL)
           F=ETOR+EVDW+ECOUL
c
c...       Calcul de la derivee premiere de l'energie au point DELTA(1).
c
           CALL CONT1(DELTA(1),ETOR1,EVDW1,ECOUL1)
           G(1)=ETOR1+EVDW1+ECOUL1
c
c...       Assignation des parametres de VF02AD.
c
           C(1)=DELTA(1)
           CN(1,1)=1.D0
           C(2)=2*PI-DELTA(1)
           CN(1,2)=-1.D0

           CALL VF02AD(N,M,MEQ,DELTA,F,G,C,CN,LCN,MAXFUN,ACC,IPRINT,INF,
        1              W,LW)

10      CONTINUE
        ENEMIN=F
c
c...    Conversion de la solution en degre.
c
        DELTA(1)=DEG(DELTA(1))
        DELTA(1)=DELTA(1)-INT(DELTA(1)/360.0)*360.0
c
c       3. Sortie des resultats.
c
        WRITE(*,1000)
        WRITE(*,1100) DELTA(1),ENEMIN
c
c...    Message a l'ecran si la solution n'est peut-etre pas optimale.
c
        IF (INF.GE.2) THEN
           WRITE(*,1950)
        ENDIF

        RETURN
```

```
c
c       Formats :
c       ---------
c
1000    FORMAT(/,1X,T5,'Solution de la minimisation de l''energie :',/)
1100    FORMAT(1X,T10,'Angle = ',F15.7,/,1X,T10,'Valeur de l''energie',
        1       ' = ',F15.7)
1200    FORMAT(F10.6)
1300    FORMAT(1X,T5,'Entrez une approximation de l''angle',
        1       ' minimisant l''energie : ')
1400    FORMAT(/,1X,T10,'Angle = ',$)
1950    FORMAT(/,1X,T5,'ATTENTION : la solution obtenue n''est peut etre'
        1       ,' pas optimale.')

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE MINFCT (RAC)
      =======================
c
c
c     Description :
c     -------------
c
c     Cette procedure calcule le minimum sans contrainte d'energie RAC
c     (degre) de la fonction objectif.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  RAC
c
c     Commons :
c     .........
c
      COMMON  /DIVISI/  NBATEG
      COMMON  /LIAISO/  AXE

      INTEGER           NBATEG,AXE(4)
c
c     Variables internes :
c     ....................
c
      DOUBLE PRECISION  RES,FIDP,FCCOS,FCSIN,PI,DEG
c
c
c     Corps :
c     -------
c
      PI=ACOS(-1.0)
c
c...  Calcul des constantes de la fonction objectif.
c
      CALL INFCT(FIDP,FCCOS,FCSIN)
c
c...  Recherche analytique de l'angle optimum.
c
      IF (FCCOS.EQ.0.D0) THEN
         RAC=PI/2
      ELSE
         RAC=ATAN (FCSIN/FCCOS)
      ENDIF

      IF ((FCCOS*COS(RAC)+FCSIN*SIN(RAC)).GT.0.D0) THEN
         RAC=RAC+PI
         IF (RAC.GT.PI) THEN
            RAC=RAC-2*PI
         ENDIF
      ENDIF
```

```
c
c...    Calcul de la valeur optimale de la fonction objectif.
c
        RES=FIDP+FCCOS*COS(RAC)+FCSIN*SIN(RAC)
        RES=SQRT(RES/NBATEG)
c
c...    Conversion de la solution en degre.
c
        RAC=DEG(RAC)
        RAC=RAC-INT(RAC/360.0)*360.0
c
c...    Impression de la solution a l'ecran.
c
        WRITE(*,1000)
        WRITE(*,1100) RAC,AXE(2),AXE(3)
        WRITE(*,1200) RES

        RETURN
c
c       Formats :
c       ---------
c
1000    FORMAT(/,1X,T5,'Solution sans contrainte d''energie :',/)
1100    FORMAT(1X,T6,F15.7,' autour de l''axe ',I2,' - ',I2)
1200    FORMAT(1X,T10,'RMS = ',F15.7,/)

        END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE MINFIT1 (ENEMIN,DELTA,DEL,F)
c     ===================
c
c     Description :
c     -------------
c
c     Cette routine minimise la somme des distances au carre entre les atomes
c     a comparer sous contrainte: l'energie doit est plus petite ou egale
c     a l'energie minimum plus une tolerance de 20 (kcal/mole).
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  ENEMIN,DELTA(1),DEL,F
c
c     Commons :
c     .........
c
      COMMON  /DIVISI/  NBATEG
      COMMON  /LIAISO/  AXE

      INTEGER           NBATEG,AXE(4)
c
c     Variables internes :
c     ....................
c
      INTEGER           LW,LCN,N,M,MEQ,MAXFUN,IPRINT,INF
      DOUBLE PRECISION  IDP,CCS,CSN,
     1                  ETOR,ETOR1,EVDW,EVDW1,ECOUL,ECOUL1,
     2                  G(1),C(3),CN(5,3),ACC,PI,W(60),
     3                  DEG,RAD,RMS
c
c
c     Corps :
c     -------
c
c
c     1. Assignation des parametres de la procedure VF02AD.
c
      PI=ACOS(-1.0)
      LW=60
      LCN=5
      N=1
      M=3
      MEQ=0
      MAXFUN=100
      ACC=1.D-03
      IPRINT=-1
      INF=-1
```

```
c
c...    Initialisation de l'angle optimal.
c
        WRITE(*,1300)
        WRITE(*,1400) AXE(2),AXE(3)
        READ (*,1200) DELTA(1)
        DELTA(1)=RAD(DELTA(1))
c
c       2. Calcul des constantes de la fonction objectif.
c
        CALL INFCT(IDP,CCS,CSN)
c
c       3. Calcul du minimum de la fonction objectif.
c
        DO 40 WHILE (INF.LE.0)
c
c...       Calcul de la fonction objectif et de sa derivee premiere.
c
           F=IDP+COS(DELTA(1))*CCS+SIN(DELTA(1))*CSN
           G(1)=-SIN(DELTA(1))*CCS+COS(DELTA(1))*CSN
c
c...       Calcul des contraintes et de leur derivee premiere.
c
           CALL CONT(DELTA(1),ETOR,EVDW,ECOUL)
           CALL CONT1(DELTA(1),ETOR1,EVDW1,ECOUL1)

           C(1)=ENEMIN+20-ETOR-EVDW-ECOUL
           CN(1,1)=-ETOR1-EVDW1-ECOUL1
           C(2)=DELTA(1)
           CN(1,2)=1
           C(3)=2*PI-DELTA(1)
           CN(1,3)=-1

           CALL VF02AD(N,M,MEQ,DELTA,F,G,C,CN,LCN,MAXFUN,ACC,
     1                 IPRINT,INF,W,LW)

40      CONTINUE
c
c...    Conversion de la solution en degre.
c
        DEL=DEG(DELTA(1))
        DEL=DEL-INT(DEL/360.0)*360.0
c
c...    Calcul du rms.
c
        RMS=SQRT(F/NBATEG)
c
c       4. Sortie des resultats.
c
        WRITE(*,1000)
        WRITE(*,1100) DEL,AXE(2),AXE(3)
        WRITE(*,1150) RMS
```

```
c
c...    Message a l'ecran si la solution n'est peut etre pas optimale.
c
        IF (INF.GE.2) THEN
           WRITE(*,1950)
        ENDIF
        CALL WAIT(20.0)


        RETURN
c
c       Formats :
c       ---------
c
1000    FORMAT(/,1X,T5,'Solution avec contrainte d''energie :',/)
1100    FORMAT(1X,T10,F15.7,' autour de l''axe ',I2,' - ',I2)
1150    FORMAT(1X,T14,'RMS = ',F15.7,/)
1200    FORMAT(F10.6)
1300    FORMAT(/,1X,T5,'Entrez une approximation de l''angle',
        1       ' optimal recherche : ',/)
1400    FORMAT(1X,T10,'Angle autour de l''axe 'I2,' - ',I2,' = ',$)
1950    FORMAT(1X,T5,'ATTENTION : la solution obtenue n''est peut etre'
        1     ,' pas optimale')

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE MINFIT2 (ZER,RAC,DELTA,DEL,FCT)
c     ==================
c
c     Description :
c     -------------
c
c     Cette routine recherche le minimum DELTA (degre), DEL (radian)
c     appartenant a l'intervalle admissible [ZER(1),ZER(2)].
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  ZER(2),RAC,DELTA,DEL,FCT
c
c     Commons :
c     .........
c
      COMMON  /DIVISI/  NBATEG
      COMMON  /LIAISO/  AXE

      INTEGER           NBATEG,AXE(4)
c
c     Variables internes :
c     ....................
c
      DOUBLE PRECISION  FIDP,FCCOS,FCSIN,F1,F2,RMS,
     1                  RAD,DEG
c
c
c     Corps :
c     -------
c
c...  Conversion de RAC en radian.
c
      RAC=RAD(RAC)
c
c...  Calcul des constantes de la fonction objectif.
c
      CALL INFCT(FIDP,FCCOS,FCSIN)
c
c     Calcul du minimum sous contrainte d'energie.
c
```

```
c
c       1. La solution sans contrainte d'energie n'est pas admissible.
c
        IF (RAC.GT.ZER(2).OR.RAC.LT.ZER(1)) THEN
c
c...       Calcul de la fonction objectif en ZER(1) et ZER(2).
c
           F1=FIDP+FCCOS*COS(ZER(1))+FCSIN*SIN(ZER(1))
           F2=FIDP+FCCOS*COS(ZER(2))+FCSIN*SIN(ZER(2))

           IF (F1.LT.F2) THEN
              DELTA=ZER(1)
              FCT=F1
           ELSE
              DELTA=ZER(2)
              FCT=F2
           ENDIF
c
c       2.  La solution sans contrainte d'energie est admissible.
c
        ELSE
           DELTA=RAC
           FCT=FIDP+FCCOS*COS(RAC)+FCSIN*SIN(RAC)
        ENDIF
c
c...    Calcul du rms RMS.
c
        RMS = SQRT(FCT/NBATEG)
c
c...    Conversion de la solution en degre.
c
        DEL=DEG(DELTA)
c
c...    Impression du resultat a l'ecran.
c
        WRITE(*,1000)
        WRITE(*,1100) DEL,AXE(2),AXE(3)
        WRITE(*,1200) RMS
        CALL WAIT(20.0)


        RETURN
c
c       Formats :
c       ---------
c
1000    FORMAT(1X,T5,'Solution sous contrainte d''energie :',/)
1100    FORMAT(1X,T6,F15.7,' autour de l''axe ',I2,' - ',I2)
1200    FORMAT(1X,T10,'RMS = ',F15.7,/)

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE ZERO (ENEMIN,DELTA)
c     ==============
c
c     Description :
c     -------------
c
c     Cette procedure recherche la racine de l'energie de conformation -
c     ENEMIN (valeur minimale de l'energie) - 20  la plus proche de zero sur
c     l'intervalle [0,PI] (DELTA(1)) et la racine la plus proche de zero sur
c     l'intervalle [-PI,0] (DELTA(2)).
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  ENEMIN,DELTA(2)
c
c     Variables internes :
c     ....................
c
      PARAMETER  EPS = 1.D-3

      INTEGER           K,MAXIT,I
      DOUBLE PRECISION  ETOR1,EVDW1,ECOUL1,
     1                  A,B,ERR,X,Y,
     2                  BORINF,BORSUP,PI
c
c
c     Corps :
c     -------
c
c...  Calcul de la valeur de PI.
c
      PI=ACOS(-1.0)
c
c...  Initialisation des parametres de la routine NBO1AD.
c
      ERR=1.D-6
      MAXIT=100
c
c     1. Recherche de la racine la plus proche de zero sur
c        l'intervalle [0,PI].
c
c...  Initialisation de la recherche.
c
      K=0
      A=0.D0
      B=PI
      DELTA(2)=PI
```

```
      DO 20 WHILE (K.LE.2)
        DO 30 WHILE (K.LE.1)
C
C...      Recherche de la racine sur l'intervalle [A,B].
C
          CALL NBO1AD(K,A,B,ERR,X,Y,MAXIT)
C
C...      Calcul de la fonction.
C
          CALL CONT (X,ETOR1,EVDW1,ECOUL1)
          Y=20+ENEMIN-ETOR1-EVDW1-ECOUL1
30      CONTINUE
C
C...    Si k vaut 2, la routine a trouve une racine sur l'intervalle [A,B].
C...    Cette racine est la plus proche de zero de toutes les racines deja
C...    rencontrees.
C...    On reduit l'intervalle [A,B] pour obtenir une racine plus proche
C...    de zero.
C
        IF (K.EQ.2) THEN
            K=0
            B=X-EPS
            DELTA(2)=X
        ENDIF
20    CONTINUE
C
C     2. Recherche de la racine la plus proche de zero sur
C        l'intervalle [-PI,0].
C
      IF (K.EQ.3) THEN
C
C...  Initialisation de la recherche.
C
        K=0
        A=-PI
        B=0.D0
        DELTA(1)=-PI
        DO 40 WHILE (K.LE.2)
          DO 50 WHILE (K.LE.1)
C
C...        Recherche de la racine sur l'intervalle [A,B].
C
            CALL NBO1AD(K,A,B,ERR,X,Y,MAXIT)
C
C...        Calcul de la fonction.
C
            CALL CONT (X,ETOR1,EVDW1,ECOUL1)
            Y=20+ENEMIN-ETOR1-EVDW1-ECOUL1
50        CONTINUE
```

```
c
c...      Si k vaut 2, la routine a trouve une racine sur l'intervalle [A,B].
c...      Cette racine est la plus proche de zero de toutes les racines deja
c...      rencontrees.
c...      On reduit l'intervalle [A,B] pour obtenir une racine plus proche
c...      de zero.
c
            IF (K.eq.2) THEN
                K=0
                A=X+EPS
                DELTA(1)=X
            ENDIF
40          CONTINUE
c
c...      Un message apparait a l'ecran si la solution n'est peut etre pas
c...      optimale.
c
            IF (K.EQ.4) THEN
              WRITE(*,1950)
            ENDIF

          ENDIF


          RETURN
c
c         Formats :
c         ---------
c
1950      FORMAT(1X,T10,'ATTENTION : la solution obtenue n''est peut etre'
          1       ,'pas optimale.')

          END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE RECEXT (Z,N)
c     ==================
c
c     Description :
c     -------------
c
c     Cette procedure recherche les racines de la derivee premiere, G, de
c     la fonction d'energie. Ces racines sont mises dans le tableau Z qui
c     comprendra, a la fin de la procedure, N elements (N >= 2).
c     HYP : deux racines sont distantes d'au moins PAS.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  MAXOP = 40

      INTEGER           N
      DOUBLE PRECISION  Z(MAXOP,2)
c
c     Fonction externe :
c     ..................
c
      EXTERNAL G

      DOUBLE PRECISION  G
c
c     Variables internes :
c     ....................
c
      DOUBLE PRECISION  DELTA1,DELTA2,ERR,PAS,
     1                  RACINE,G1,G2


c
c     Corps :
c     -------
c
c...  Initalisation.
c
      ERR=1.D-6
      PAS=10.D0

      N=1
      Z(1,1)=0.D0
      DELTA1=0.D0
      DELTA2=DELTA1+PAS
      G1=G(DELTA1)
      G2=G(DELTA2)
```

```
C
C...    On recherche les racines sur [0,360].
C
        DO 10 WHILE(DELTA1.LT.360.D0)
C
C...       Si DELTA1 et DELTA2 ne sont pas racines de G...
C
           IF ((DABS(G1).GT.ERR).AND.(DABS(G2).GT.ERR)) THEN
C
C...          alors si G(DELTA1) et G(DELTA2) sont de signes opposes,
C...          il existe une racine dans l'intervalle ]DELTA1,DELTA2[.
C
              IF (G1*G2.LT.0.D0) THEN
                 N=N+1
                 Z(N,1)=RACINE(DELTA1,PAS,G)
              ENDIF
C
C...       Sinon si G(DELTA2)=0, alors DELTA2 est racine.
C
           ELSEIF (DABS(G2).LE.ERR) THEN
              N=N+1
              Z(N,1)=DELTA2
           ENDIF

           DELTA1=DELTA2
           DELTA2=DELTA1+PAS
           G1=G2
           G2=G(DELTA2)

10      CONTINUE

        IF (Z(N,1).NE.360.D0) THEN
           N=N+1
           Z(N,1)=360.D0
        ENDIF


        RETURN
        END
C
C -------------------------------------------------------------------------
C -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE MINMAX (Z,N)
      ==================
c
c
c     Description :
c     -------------
c
c     Cette procedure retire les eventuels points d'inflexion du tableau
c     des extrema Z, et differencie les extrema.
c
c     Declarations :
c     --------------
c
      PARAMETER  MAXOP = 40

      INTEGER           N
      DOUBLE PRECISION  Z(MAXOP,2)
c
c
c     Variables internes :
c     --------------------
c
      INTEGER           K,L
      DOUBLE PRECISION  X,VAL,ERR,ETOR2,ECOUL2,EVDW2,
     1                  RAD
c
c
c     Corps :
c     -------
c
      ERR=1.D-5
      L=2

      DO 20 WHILE (L.LE.N-1)
c
c...     Calcul de la derivee seconde des energies de torsion, de Van Der
c...     Waals et de Coulomb pour un angle de X radian.
c
         X=RAD(Z(L,1))
         CALL CONT2(X,ETOR2,EVDW2,ECOUL2)
         VAL=ETOR2+ECOUL2+EVDW2
c
c...     Si au point X, la derivee seconde est strictement positive, alors X
c...     (i.e. Z(L,1)) est un minimum de la fonction d'energie.
c
         IF ((DABS(VAL).GT.ERR).AND.(SIGN(1.DO,VAL).EQ.+1)) THEN
            Z(L,2)=-1.0
            L=L+1
c
c...     Sinon si au point X, la derivee seconde est strictement negative,
c...     alors X (i.e. Z(L,1)) est un maximum de la fonction d'energie.
c
         ELSEIF ((DABS(VAL).GT.ERR).AND.(SIGN(1.DO,VAL).EQ.-1)) THEN
            Z(L,2)=+1.0
            L=L+1
```

```
c
c...        Sinon X est un point d'inflexion. On le retire donc de la liste Z.
c
            ELSE
               DO 10 K=L+1,N
                  Z(K-1,1)=Z(K,1)
10             CONTINUE
               N=N-1
            ENDIF
20       CONTINUE


         RETURN
         END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE RECOPT (Z,N,DAUX,DELTA,DEL,SOM)
c     ==================
c
c
c     Description :
c     -------------
c
c     Cette procedure recherche le meilleur angle de rotation, DELTA (radian)
c     ou DEL (degre) pour effectuer le FITTING, avec contraintes d'energie.
c     SOM est la valeur de la fonction objectif pour cet angle DELTA.
c     DAUX1 est le meilleur angle (radian) de rotation sans contrainte
c     d'energie.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  MAXOP = 40

      INTEGER           N
      DOUBLE PRECISION  Z(MAXOP,2),DAUX,DELTA,DEL,SOM
c
c     Fonction externe :
c     ..................
c
      EXTERNAL Y

      DOUBLE PRECISION  Y
c
c     Commons :
c     .........
c
      COMMON  /UN/      A
      COMMON  /DIVISI/  NBATEG
      COMMON  /LIAISO/  AXE

      INTEGER           NBATEG,AXE(4)
      DOUBLE PRECISION  A
c
c     Variables internes :
c     ....................
c
      INTEGER           L,SORTIE
      DOUBLE PRECISION  B,D1,D2,RMS,
     1                  E,RACINE,F,DEG,RAD
c
c
c     Corps :
c     -------
c
c...  Recherche du meilleur angle de rotation avec contraintes d'energie, D1,
c...  en partant de 0 vers 360.
c
      L=1
      SORTIE=0
```

```
      DO 10 WHILE (SORTIE.EQ.0)
         A=Z(L,1)
         B=Z(L+1,1)
C
C...     Si l'angle de rotation DAUX appartient a l'intervalle [A,B]...
C
         IF (DAUX.LE.B) THEN
C
C...        alors si cet angle est "accessible", nous l'identifions a
C...        l'angle D1 recherche...
C
            IF (E(DAUX).LE.E(A)+20.D0) THEN
               SORTIE=1
C
C...        sinon, D1 est l'angle correspondant a l'energie E(A)+20.
C
            ELSE
               SORTIE=2
            ENDIF
C
C...     Si DAUX n'appartient pas a [A,B], nous devons considerer
C        l'intervalle suivant si il est "accessible" :
C
         ELSE
C
C...        si B est maximum de la fonction d'energie, nous devons verifier
C...        que nous pouvons atteindre ce point.
C
            IF (Z(L+1,2).EQ.1.D0) THEN
C
C...           si B n'est pas "accessible", alors D1 est l'angle correspon-
C...           dant a l'energie E(A)+20.
C
               IF (E(B).GT.E(A)+20.D0) THEN
                  SORTIE=3
C
C...           sinon l'intervalle suivant est "accessible".
C
               ELSE
                  L=L+1
               ENDIF
            ELSE
               L=L+1
            ENDIF
         ENDIF
10    CONTINUE

      IF (SORTIE.EQ.1) THEN
         D1=DAUX
      ELSEIF (SORTIE.EQ.2) THEN
         D1=RACINE(A,DAUX-A,Y)
      ELSE
         D1=RACINE(A,B-A,Y)
      ENDIF
```

```
c
c...    Recherche du meilleur angle de rotation avec contraintes d'energie, D2,
c...    en partant de 360 vers 0.
c
        L=N
        SORTIE=0
        DO 20 WHILE (SORTIE.EQ.0)
           A=Z(L,1)
           B=Z(L-1,1)
           IF (DAUX.GE.B) THEN
              IF (E(DAUX).LE.E(A)+20.DO) THEN
                 SORTIE=1
              ELSE
                 SORTIE=2
              ENDIF
           ELSE
              IF (Z(L-1,2).EQ.1.DO) THEN
                 IF (E(B).GT.E(A)+20.DO) THEN
                    SORTIE=3
                 ELSE
                    L=L-1
                 ENDIF
              ELSE
                 L=L-1
              ENDIF
           ENDIF
20      CONTINUE

        IF (SORTIE.EQ.1) THEN
           D2=DAUX
        ELSEIF (SORTIE.EQ.2) THEN
           D2=RACINE(A,DAUX-A,Y)
        ELSE
           D2=RACINE(A,B-A,Y)
        ENDIF
        D2=D2-360.0
c
c...    L'angle DELTA recherche est le meilleur des deux angles D1, D2.
c
        IF (D1.NE.D2) THEN
           IF (F(D1).LE.F(D2)) THEN
              DELTA=D1
           ELSE
              DELTA=D2
           ENDIF
        ELSE
           DELTA=D1
        ENDIF
c
c...    Calcul de SOM et du rms RMS.
c
        SOM = F(DELTA)
        RMS = SQRT(SOM/NBATEG)
c
c...    DEL est l'angle optimal exprime en degre (compris entre 0 et 360).
c...    DELTA doit etre exprime en radian.
c
        DEL=DELTA
        DELTA=RAD(DELTA)
```

```
c
c...    Impression du resultat a l'ecran.
c
        WRITE(*,1000)
        WRITE(*,1100) DEL,AXE(2),AXE(3)
        WRITE(*,1200) RMS
        CALL WAIT(20.0)

        RETURN
c
c       Formats :
c       ---------
c
1000    FORMAT(//,1X,T5,'Solution sous contrainte d''energie :',/)
1100    FORMAT(1X,T6,F15.7,' autour de l''axe ',I2,' - ',I2)
1200    FORMAT(1X,T10,'RMS = ',F15.7,/)

        END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      DOUBLE PRECISION FUNCTION RACINE (THETA,PAS,G)
      ==============================
c
c
c     Description :
c     -------------
c
c     Cette fonction retourne la valeur de LA racine de la fonction G situee
c     dans ]THETA,THETA+PAS[ si PAS > 0, dans ]THETA+PAS,THETA[ si PAS < 0.
c     La methode utilisee est celle du balayage.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  THETA,PAS,G
c
c     Variables internes :
c     ....................
c
      DOUBLE PRECISION  B1,B2,G1,G2,ERR,PAS1


      PARAMETER  P = 4
c
c
c     Corps :
c     -------
c
c...  Initialisations.
c
      ERR=1.D-6
      PAS1= PAS/P

      B1=THETA
      B2=THETA+PAS1
      G1=G(B1)
      G2=G(B2)
```

```
c
c...    Tant que nous n'avons pas trouve la racine, nous continuons le balayage
c
        DO 10 WHILE (DABS(G2).GT.ERR)
           IF ((G1*G2).LT.0.D0) PAS1=-PAS1/P
           B1=B2
           B2=B2+PAS1
           G1=G2
           G2=G(B2)
10      CONTINUE

        RACINE=B2


        RETURN
        END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      DOUBLE PRECISION FUNCTION G (DELTA)
c     ===========================
c
c     Description :
c     -------------
c
c     Cette fonction retourne la valeur de la derivee premiere de la fonction
c     d'energie pour un angle DELTA (degre).
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      DOUBLE PRECISION  DELTA
c
c     Variables internes :
c     ....................
c
      DOUBLE PRECISION  DELTA1,
     1                  ETOR1,EVDW1,ECOUL1,
     2                  RAD
c
c
c     Corps :
c     -------
c
c...  La procedure CONT1 donne la valeur de la derivee premiere des energies
c...  de torsion, de Van Der Waals et de Coulomb pour un angle DELTA1
c...  exprime en RADIAN.
c
      DELTA1=RAD(DELTA)
      CALL CONT1(DELTA1,ETOR1,EVDW1,ECOUL1)
      G=ETOR1+EVDW1+ECOUL1


      RETURN
      END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
C ------------------------------------------------------------------------------
C ------------------------------------------------------------------------------
C
      DOUBLE PRECISION FUNCTION Y (THETA)
      ===========================
C
C
C     Description :
C     -------------
C
C     Cette fonction prend la valeur de la fonction E(theta)-E(a)-20 ou THETA
C     est un angle exprime en degre.
C
C
C     Declarations :
C     --------------
C
C     Parametre :
C     ...........
C
      DOUBLE PRECISION  THETA
C
C     Common :
C     ........
C
      COMMON  /UN/  A

      DOUBLE PRECISION  A
C
C     Variable interne :
C     ..................
C
      DOUBLE PRECISION  E
C
C
C     Corps :
C     -------
C
      Y=E(THETA)-E(A)-20.D0

      RETURN
      END
C
C ------------------------------------------------------------------------------
C ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      DOUBLE PRECISION FUNCTION E (THETA)
c     ===========================
c
c     Description :
c     -------------
c
c     Cette fonction prend la valeur de l'energie pour un angle THETA
c     exprime en degre.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      DOUBLE PRECISION  THETA
c
c     Variables internes :
c     ....................
c
      DOUBLE PRECISION  THETA1,
     1                  ETOR,EVDW,ECOUL,
     2                  RAD
c
c
c     Corps :
c     -------
c
c...  La procedure CONT donne la valeur des energies de torsion, de Coulomb
c...  et de Van Der Waals pour un angle DELTA1 exprime en RADIAN.
c
      THETA1=RAD(THETA)
      CALL CONT(THETA1,ETOR,EVDW,ECOUL)
      E=ETOR+EVDW+ECOUL


      RETURN
      END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
C ------------------------------------------------------------------------------
C ------------------------------------------------------------------------------
C
      DOUBLE PRECISION FUNCTION F (THETA)
      ===========================
C
C
C     Description :
C     -------------
C
C     Cette fonction prend la valeur de la fonction objectif pour un angle
C     THETA exprime en degre.
C
C
C     Declarations :
C     --------------
C
C     Parametre :
C     ...........
C
      DOUBLE PRECISION  THETA
C
C     Variables internes :
C     ....................
C
      DOUBLE PRECISION  THETA1,
     1                  FIDP,FCCOS,FCSIN,
     2                  RAD
C
C
C     Corps :
C     -------
C
      THETA1=RAD(THETA)
      CALL INFCT(FIDP,FCCOS,FCSIN)
      F=FIDP+COS(THETA1)*FCCOS+SIN(THETA1)*FCSIN


      RETURN
      END
C
C ------------------------------------------------------------------------------
C ------------------------------------------------------------------------------
```

```
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
c
      SUBROUTINE MINI (DELTA,DEL)
c     ===============
c
c     Description :
c     -------------
c
c     Cette procedure calcule le minimum (DELTA en radian et DEL en degre)
c     de la somme des distances au carre entre les atomes a comparer.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NAXE = 10
      PARAMETER  FMAX = 3*NAXE*NAXE+8*NAXE

      DOUBLE PRECISION  DELTA(NAXE),DEL(NAXE)
c
c     Commons :
c     .........
c
      COMMON  /DIVISI/  NBATEG
      COMMON  /LIAISO/  NBRAXE,AXE

      INTEGER           NBATEG,NBRAXE,AXE(4,NAXE)
c
c     Variables internes :
c     ....................
c
      INTEGER           LWA,IWA(NAXE),INFO,I
      DOUBLE PRECISION  FVEC(3*NAXE),TOL,WA(FMAX),FNORM,
      1                 DPMPAR,ENORM,RAD,DEG
      EXTERNAL  FONCT
c
c
c     Corps :
c     .......
c
c
c...  Initialisation d'un parametre de travail de la routine d'optimisation.
c
      LWA=NBATEG*NBRAXE+5*NBRAXE+NBATEG
c
c...  Met la tolerance TOL a la racine carree de la precision machine.
c
      TOL=DSQRT(DPMPAR(1))
```

```
c
c...    Initialisation des angles de rotation.
c
        WRITE(*,1100)
        WRITE(90,2000) NBRAXE
        DO 10 I=1,NBRAXE

           WRITE(*,1150) I,AXE(2,I),AXE(3,I)
           READ(*,1000) DELTA(I)
c
c...       Impression des angles initiaux sur le fichier des iterees
c
           WRITE(90,2100) DELTA(I),AXE(2,I),AXE(3,I)
           DELTA(I)=RAD(DELTA(I))

10      CONTINUE
c
c
c...    Cette routine minimise la somme des carres des distances entre les
c...    atomes a egaler. Le tableau DELTA contient les angles optimaux et
c...    le tableau FVEC contient les distances au carre des differents atomes
c...    a egaler.
c
        CALL LMDIF1 (FONCT,NBATEG,NBRAXE,DELTA,FVEC,TOL,INFO,IWA,WA,LWA)
c
c...    Conversion des angles optimaux en degre compris entre 0 et 360.
c
        DO 20 I=1,NBRAXE
           DEL(I)=DEG(DELTA(I))
           DEL(I)=DEL(I)-INT(DEL(I)/360.0)*360.0
20      CONTINUE
c
c...    Calcul du RMS (ENORM calcule la norme euclidienne du vecteur FVEC).
c
        FNORM = ENORM(NBATEG,FVEC)
        FNORM = SQRT(FNORM/NBATEG)
c
c...    Impression des resultats a l'ecran.
c
        WRITE(*,1200)
        WRITE(*,1300) (DEL(I),AXE(2,I),AXE(3,I),I=1,NBRAXE)
        WRITE(*,1400) FNORM
        CALL WAIT(20.0)
c
c...    Message a l'ecran si la solution n'est peut-etre pas optimale.
c
        IF (INFO.EQ.5) THEN
           WRITE(*,1650)
        ENDIF


        RETURN
```

```
c
c       Formats :
c       ---------
c
1000    FORMAT(F10.6)
1100    FORMAT(/,1X,T5,'Entrez une approximation de chaque angle de',
        1       ' rotation : ',/)
1150    FORMAT(1X,T10,'Angle(',I2,') autour de l''axe 'I2,' - ',I2,
        1       ' = ',$)
1200    FORMAT(/,1X,T5,'Solutions de la minimisation :',/)
1300    FORMAT(1X,T6,F15.7,' autour de l''axe ',I2,' - ',I2)
1400    FORMAT(1X,T10,'RMS = ',F15.7,/)
1650    FORMAT(/,1X,T5,'ATTENTION : la solution obtenue n''est peut-etre'
        1       ,' pas optimale.')

2000    FORMAT(/,1X,T5,'Les ',I2,' angles de rotation initiaux de la',
        1       ' minimisation :',/,1X,T4,57('-'))
2100    FORMAT(1X,T20,F10.5,'  autour de  ',I2,' - ',I2)

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE FONCT (NFCT,NVAR,DELTA,FVEC,IFLAG)
c     ================
c
c     Description :
c     -------------
c
c     Cette routine evalue les valeurs des NFCT carre des distances (FVEC)
c     apres une rotation autour des NVAR axes d'un angle DELTA.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NAXE = 10

      INTEGER           NFCT,NVAR,IFLAG
      DOUBLE PRECISION  DELTA(NAXE),FVEC(3*NAXE)
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /FIXE/    XAT,X
      COMMON  /MOBILE/  YAT,Y
      COMMON  /DIVISI/  NBATEG,AME,AFE
      COMMON  /LIAISO/  NBRAXE,AXE

      INTEGER           XAT,YAT,NBRAXE,AXE(4,NAXE),
     1                  NBATEG,AME(0:NDIM1,0:NAXE),AFE(NDIM1)
      DOUBLE PRECISION  X(NDIM1,3),Y(NDIM1,3)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
      DOUBLE PRECISION  CD(3),T(3),Z(NDIM1,3),ROT(3,3),B(3)
```

```
c
c       Corps :
c       -------
c
c
c...    Transfert des coordonnees des atomes definissant les axes et des atomes
c...    a egaler dans la matrice Z.
c
        DO 20 I=1,NVAR
           DO 10 J=2,3
              Z(AXE(J,I),1)=Y(AXE(J,I),1)
              Z(AXE(J,I),2)=Y(AXE(J,I),2)
              Z(AXE(J,I),3)=Y(AXE(J,I),3)
10         CONTINUE
20      CONTINUE

        DO 30 I=1,NFCT
          Z(AME(I,0),1)=Y(AME(I,0),1)
          Z(AME(I,0),2)=Y(AME(I,0),2)
          Z(AME(I,0),3)=Y(AME(I,0),3)
30      CONTINUE

        DO 70 I=1,NVAR
c
c...      Ces routines calculent les cosinus directeurs du Ieme axe,
c...      ainsi que la translation deplacant l'axe a l'origine.
c
          CALL COSDIR(Z,AXE(1,I),CD)
          CALL TRANSF(Z,AXE(1,I),T)
c
c...      Translation des coordonnees
c
          DO 40 J=1,AME(0,I)
            Z(AME(J,I),1)=Z(AME(J,I),1)-T(1)
            Z(AME(J,I),2)=Z(AME(J,I),2)-T(2)
            Z(AME(J,I),3)=Z(AME(J,I),3)-T(3)
40        CONTINUE
c
c...      Calcul de la rotation d'un angle DELTA(i) autour de l'axe I.
c
          CALL MATROT(DELTA(I),CD,ROT)

c
c...      Rotation des atomes.
c
          DO 50 J=1,AME(0,I)
             B(1)=ROT(1,1)*Z(AME(J,I),1)+ROT(1,2)*Z(AME(J,I),2)
     +           +ROT(1,3)*Z(AME(J,I),3)
             B(2)=ROT(2,1)*Z(AME(J,I),1)+ROT(2,2)*Z(AME(J,I),2)
     +           +ROT(2,3)*Z(AME(J,I),3)
             B(3)=ROT(3,1)*Z(AME(J,I),1)+ROT(3,2)*Z(AME(J,I),2)
     +           +ROT(3,3)*Z(AME(J,I),3)
             Z(AME(J,I),1)=B(1)
             Z(AME(J,I),2)=B(2)
             Z(AME(J,I),3)=B(3)
50        CONTINUE
```

```
c
c...      Translation inverse des atomes.
c
          DO 60 J=1,AME(0,I)
             Z(AME(J,I),1)=Z(AME(J,I),1)+T(1)
             Z(AME(J,I),2)=Z(AME(J,I),2)+T(2)
             Z(AME(J,I),3)=Z(AME(J,I),3)+T(3)
60        CONTINUE

70      CONTINUE
c
c...    Calcul des carres des distances des atomes a egaler apres rotations.
c
        DO 80 I=1,NFCT
           FVEC(I)=(X(AFE(I),1)-Z(AME(I,0),1))**2
     +            +(X(AFE(I),2)-Z(AME(I,0),2))**2
     +            +(X(AFE(I),3)-Z(AME(I,0),3))**2
80        CONTINUE

        RETURN
        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      LOGICAL FUNCTION LIAD (AXE)
c     ===========================
c
c     Description :
c     -------------
c
c     Cette fonction prend la valeur vraie si la liaison autour de laquelle
c     on effectue le FITTING est admissible i.e. si la liaison est simple et
c     si elle ne fait partie d'aucun cycle; fausse sinon.
c     On verifiera egalement que l'angle diedre est bien defini.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      INTEGER           AXE(4)
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      COMMON  /MOBILE/      YAT,Y,YLIEN
      COMMON  /MOLECU/      BOND,NBAT,ITYP
      COMMON  /ADJACENCE/   ADJL
      COMMON  /CYCLE/       ARETE

      INTEGER           YAT,YLIEN,BOND(2,NDIM2),NBAT(NDIM1),ITYP(NDIM1),
      1                 ADJL(0:4,NDIM1)
      DOUBLE PRECISION  Y(NDIM1,3)
      LOGICAL           ARETE(NDIM1,NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER           ATYP(9),I,J,K
      LOGICAL           OK,ANGAD(3),TYP(2)
      CHARACTER         REP
c
c
      DATA (ATYP(I),I=1,9) / 2,3,4,7,8,9,11,13,23 /
```

```
c
c
c       Corps :
c       -------
c
        OK=.TRUE.
c
c       1. Verification de l'existence des liaisons formant l'angle diedre.
c       ...................................................................
c
        DO 20 K=1,3
           ANGAD(K)=.FALSE.
           J=1
c
c...    ANGAD(K) est mis a "vrai" si l'atome AXE(K) est lie a l'atome AXE(K+1).
c
           DO 10 WHILE ((.NOT.ANGAD(K)).AND.(J.LE.ADJL(0,AXE(K))))
              IF (ADJL(J,AXE(K)).EQ.AXE(K+1)) ANGAD(K)=.TRUE.
              J=J+1
10         CONTINUE

20      CONTINUE

        IF ((.NOT.ANGAD(1)).OR.(.NOT.ANGAD(2)).OR.(.NOT.ANGAD(3))) THEN
           OK=.FALSE.
           CALL SCLEAR
           WRITE(*,1050)
        ELSE
c
c       2. Verification du type de la liaison.
c       ......................................
c
c...       Si le type des atomes formant la liaison autour de laquelle on ef-
c...       fectue le fitting est repris dans la liste ATYP, alors IL SE PEUT
c...       que la liaison soit double ou triple.
c
           TYP(1)=.FALSE.
           TYP(2)=.FALSE.
           I=1

           DO 30 WHILE((.NOT.TYP(1).OR..NOT.TYP(2)).AND.I.LE.9)
              IF (ITYP(AXE(2)).EQ.ATYP(I)) TYP(1)=.TRUE.
              IF (ITYP(AXE(3)).EQ.ATYP(I)) TYP(2)=.TRUE.
              I=I+1
30         CONTINUE

           IF((TYP(1).EQ..TRUE.).AND.(TYP(2).EQ..TRUE.)) THEN
              CALL SCLEAR
              WRITE(*,1100)
              WRITE(*,1200)
              READ(*,1000) REP
              IF ((REP.EQ.'O').OR.(REP.EQ.'o')) OK=.FALSE.
           ENDIF


        ENDIF
```

```
      IF (OK) THEN
C
C     3. Verification de la non-appartenance de la liaison a un cycle.
C     ................................................................
C
            OK=.NOT.ARETE(AXE(2),AXE(3))
            IF (.NOT.OK) THEN
               CALL SCLEAR
               WRITE(*,1300)
            ENDIF

      ENDIF


      LIAD=OK


      RETURN
C
C     Formats :
C     ---------
C
1000  FORMAT(A)
1050  FORMAT(1X,79('.'),/,1X,'.',T30,'A T T E N T I O N',T80,'.',
     1      /,1X,'.',T9,'La liaison autour de laquelle on effectue',
     2      ' le FITTING n''existe pas',T80,'.',/,1X,79('.'),/)
1100  FORMAT(1X,79('.'),/,1X,'.',T30,'A T T E N T I O N',T80,'.',
     1      /,1X,'.',T7,'Il semble que la liaison autour de laquelle',
     2      ' on effectue le FITTING',T80,'.',/,1X,'.',T7,'soit une',
     3      ' liaison double ou triple ! ! !',T80,'.',/,1X,79('.'),/)
1200  FORMAT(1X,'Est-ce reellement le cas ?     ',$)
1300  FORMAT(1X,79('.'),/,1X,'.',T30,'A T T E N T I O N',T80,'.',
     1      /,1X,'.',T4,'La liaison autour de laquelle on effectue',
     2      ' le FITTING appartient a un cycle',T80,'.',/,1X,
     3      79('.'),/)

      END
C
C -------------------------------------------------------------------------
C -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE PALMIER (NSO,NAR,BOND,X)
c     ===================
c
c     Description :
c     -------------
c
c     Cette procedure engendre un palmier a partir du multigraphe simplement
c     connexe dont NSO est le nombre de sommets, NAR le nombre d'aretes, BOND
c     l'ensemble des paires de sommets formant les aretes. Rappelons que les
c     sommets de ce multigraphe sont adjacents a, au plus, 4 autres sommets.
c     La methode utilisee est celle developpee par R. Tarjan dans l'article
c     "Depth-first search and linear graph algorithms" paru dans Siam J.
c     Comput.(vol.1n. 2 juin 1972).
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      INTEGER NSO,NAR,BOND(2,NDIM2),X
c
c     Commons :
c     ---------
c
      COMMON  /ADJACENCE/  ADJL,PALM,NUM

      INTEGER  ADJL(0:4,NDIM1),PALM(2,NDIM2),NUM(NDIM1)
c
c     Variables internes :
c     --------------------
c
      INTEGER  U,V,W,I,J,T,N,K,
     1         PILE(3,NDIM2)
c
c
c     Corps :
c     -------
c
      DO 20 J=1,NDIM2
         DO 10 I=1,3
            PILE(I,J)=0
10       CONTINUE
20    CONTINUE

      DO 40 J=0,NDIM2
         DO 30 I=1,2
            PALM(I,J)=0
30       CONTINUE
40    CONTINUE
```

```
      DO 50 I=1,NSO
         NUM(I)=0
50    CONTINUE


      U=0
      V=X
      N=0
      T=0
      I=1
      NUM(V)=I

      CALL MAZDA (U,V,PILE,N,W)

90    IF (NUM(W).EQ.0) THEN

         I=I+1
         NUM(W)=I
         T=T+1
         PALM(1,T)=V
         PALM(2,T)=W
         U=V
         V=W
         CALL MAZDA (U,V,PILE,N,W)

      ELSEIF ((NUM(W).LT.NUM(V)).AND.(W.NE.U)) THEN

         T=T+1
         PALM(1,T)=V
         PALM(2,T)=W
         U=PILE(1,N)
         V=PILE(2,N)
         W=PILE(3,N)
         N=N-1

      ELSE

         U=PILE(1,N)
         V=PILE(2,N)
         W=PILE(3,N)
         N=N-1

      ENDIF

      IF (T.LT.NAR) GOTO 90


      RETURN
      END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE MAZDA (U,V,PILE,N,W)
c     ===============
c
c     Description :
c     -------------
c
c     Cette procedure met dans PILE les renseignements suivants :
c     V : sommet que l'on examine,
c     U : sommet precedant V,
c     W : sommet(s) suivant V.
c     A la fin de la procedure, W sera le dernier sommet suivant V contenu
c     dans la liste d'adjacence ADJL. Ce dernier sommet n'est pas stocke dans
c     PILE : c'est le prochain sommet que nous allons examiner.
c     N est une variable pointant vers le dernier 'niveau' de PILE.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      INTEGER  U,V,PILE(3,NDIM2),N,W
c
c     Common :
c     ........
c
      COMMON  /ADJACENCE/  ADJL

      INTEGER  ADJL(0:4,NDIM1)
c
c     Variable interne :
c     ..................
c
      INTEGER  I
```

```
c
c
c       Corps :
c       -------
c
        I=2

        DO 10 WHILE ((ADJL(I,V).NE.0).AND.(I.LE.4))
           N=N+1
           PILE(1,N)=U
           PILE(2,N)=V
           PILE(3,N)=ADJL(I-1,V)
           I=I+1
10      CONTINUE

        W=ADJL(I-1,V)


        RETURN
        END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      LOGICAL FUNCTION VERDEG (NSO,NAR,BOND)
c     ======================
c
c     Description :
c     -------------
c
c     Cette fonction prend la valeur "vrai" si les sommets du mutigraphe sont
c     de degre au plus quatre. Sinon, elle prend la valeur "faux". Le multi-
c     graphe est d'ordre NSO, possede NAR aretes definies par les paires de
c     sommets contenues dans BOND.
c     Cette fonction cree la structure d'adjacence ADJL du multigraphe.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM2 = 200

      INTEGER  NSO,NAR,BOND(2,NDIM2)
c
c     Common :
c     ........
c
      PARAMETER  NDIM1 = 100

      COMMON  /ADJACENCE/  ADJL,PALM,NUM

      INTEGER  ADJL(0:4,NDIM1),PALM(2,NDIM2),NUM(NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER  I,J,K
      LOGICAL  OK
```

```
c
c
c       Corps :
c       -------
c
        DO 20 J=1,NSO
           DO 10 I=0,4
              ADJL(I,J)=0
10         CONTINUE
20      CONTINUE

        OK=.TRUE.
        I=1

        DO 30 WHILE (OK.AND.I.LE.NAR)
c
c...       Si le sommet BOND(1,I) est deja lie a quatre autres sommets, VERDEG
c...       doit etre mis a "faux".
c
           IF (ADJL(0,BOND(1,I)).EQ.4) THEN
              OK=.FALSE.
c
c...       Sinon on complete la liste d'adjacence du sommet BOND(1,I).
c
           ELSE
              ADJL(0,BOND(1,I))=ADJL(0,BOND(1,I))+1
              J=ADJL(0,BOND(1,I))
              ADJL(J,BOND(1,I))=BOND(2,I)
c
c...          Si le sommet BOND(2,I) est deja lie a quatre autres sommets,
c...          VERDEG doit etre mis a "faux".
c
              IF (ADJL(0,BOND(2,I)).EQ.4) THEN
                 OK=.FALSE.
c
c...          Sinon on complete la liste d'adjacence du sommet BOND(2,I).
c
              ELSE
                 ADJL(0,BOND(2,I))=ADJL(0,BOND(2,I))+1
                 K=ADJL(0,BOND(2,I))
                 ADJL(K,BOND(2,I))=BOND(1,I)
              ENDIF
           ENDIF
           I=I+1
30      CONTINUE

        VERDEG=OK


        RETURN
        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      LOGICAL FUNCTION VERSC (NSO,NAR,BOND)
      =====================
c
c     Description :
c     -------------
c
c     Cette fonction prend la valeur "vrai" si le multigraphe d'ordre NSO
c     dont NAR est le nombre d'aretes et BOND l'ensemble des paires de som-
c     mets definissant les aretes, est simplement connexe; sinon elle prend
c     la valeur "faux".
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM2 = 200

      INTEGER  NSO,NAR,BOND(2,NDIM2)
c
c     Common :
c     ........
c
      PARAMETER  NDIM1 = 100

      COMMON  /ADJACENCE/  ADJL,PALM,NUM

      INTEGER  ADJL(0:4,NDIM1),PALM(2,NDIM2),NUM(NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER  I
      LOGICAL  OK
```

```
c
c
c       Corps :
c       -------
c
        OK=.TRUE.
c
c...    On engendre un palmier a partir du multigraphe.
c
        CALL PALMIER(NSO,NAR,BOND,BOND(1,1))

        I=1
        DO 10 WHILE(OK.AND.(I.LE.NSO))
c
c...       Si il existe au moins un sommet non numerote alors le multigraphe
c...       n'est pas simplement connexe.
c
           IF (NUM(I).EQ.0) OK=.FALSE.
           I=I+1

10      CONTINUE

        VERSC=OK


        RETURN
        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      LOGICAL FUNCTION VERMS (NSO)
      ============================
c
c     Description :
c     -------------
c
c     Cette fonction prend la valeur "vrai" si le multigraphe d'ordre NSO et
c     de structure d'adjacence ADJL est un multigraphe simple; "faux" sinon.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      INTEGER  NSO
c
c     Common :
c     ........
c
      PARAMETER  NDIM1 = 100

      COMMON  /ADJACENCE/  ADJL

      INTEGER  ADJL(0:4,NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER  I,J,K
      LOGICAL  OK
```

```
c
c
c       Corps :
c       -------
c
        OK=.TRUE.
        K=1
c
c...    Pour la liste d'adjacence de chaque sommet :
c
        DO 30 WHILE((K.LE.NSO).AND.OK)
c
c...       On verifie que le sommet K n'est pas lie a lui-meme i.e. que le mul-
c...       tigraphe n'a pas de boucle.
c
           IF (ADJL(1,K).EQ.K) OK=.FALSE.
           J=2
           DO 20 WHILE((J.LE.ADJL(0,K)).AND.OK)
c
c...          On verifie que le sommet K n'est pas lie a lui-meme i.e. que
c...          le multigraphe n'a pas de boucle.
c
              IF (ADJL(J,K).EQ.K) OK=.FALSE.
c
c...          On verifie que le sommet K n'est pas lie deux fois au meme sommet
c...          i.e. que le multigraphe est un 1-multigraphe.
c
              DO 10 I=J-1,1,-1
                 IF (ADJL(J,K).EQ.ADJL(I,K)) OK=.FALSE.
10            CONTINUE

              J=J+1

20         CONTINUE

           K=K+1

30      CONTINUE


        VERMS=OK


        RETURN
        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
c
      SUBROUTINE PRECH (X,Y)
c     ==========
c
c     Description :
c     -------------
c
c     Cette procedure recherche l'ascendant du sommet X dans l'arborescence
c     constituant l'extremite d'une chaine passant par X, chaine dont
c     les aretes font partie d'un cycle. Le sommet Y est le suivant de
c     cet ascendant, suivant appartenant au chemin reliant l'ascendant a X.
c     L'ascendant est encore note X.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      INTEGER  X,Y
c
c     Common :
c     ........
c
      PARAMETER  NDIM1 = 100

      COMMON  /PRECEDENT/  PREC

      INTEGER  PREC(2,NDIM1)
c
c
c     Corps :
c     -------
c
c...  Tant que le sommet courant X possede un precedent deja examine, nous
c...  nous positionnons sur ce precedent.
c
      DO 10 WHILE (PREC(1,X).NE.0)
         Y=X
         X=PREC(1,X)
10    CONTINUE


      RETURN
      END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE CHAINE (Y,K)
c     ==================
c
c     Description :
c     -------------
c
c     Cette procedure parcourt la chaine de l'arborescence de PALM partant
c     du sommet PALM(1,K) et aboutissant au sommet Y. Toutes les aretes de
c     cette chaine font alors partie d'un cycle.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      INTEGER  Y,K
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      COMMON /ADJACENCE/  ADJL,PALM,NUM
      COMMON /CYCLE/      ARETE
      COMMON /PRECEDENT/  PREC

      INTEGER  ADJL(0:4,NDIM1),PALM(2,NDIM2),NUM(NDIM1),PREC(2,NDIM1)
      LOGICAL  ARETE(NDIM1,NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER  X,X1,X2,Y1,N
```

```
c
c
c       Corps :
c       -------
c
        N=K
        X=PALM(1,N)
c
c...    Tant que le sommet courant n'est pas le sommet Y ou un de ses ascen-
c...    dants dans l'arborescence deja examines :
c
        DO 10 WHILE (NUM(X).GT.NUM(Y))

           N=N-1
           X2=PALM(2,N)

           IF (X2.EQ.X) THEN
              X1=PALM(1,N)
              ARETE(X1,X2)=.TRUE.
              ARETE(X2,X1)=.TRUE.
              PREC(1,X)=X1
              PREC(2,X)=N
              X=X1
c
c...          Si le nouveau sommet X fait partie d'une chaine deja parcourue,
c...          nous nous positionnons a l'extremite de cette chaine.
c
              IF (PREC(1,X).NE.0) THEN
                 CALL PRECH(X,Y1)
                 N=PREC(2,Y1)
              ENDIF

           ENDIF

10      CONTINUE


        RETURN
        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE REPCY (NSO,NAR,BOND)
c     ===============
c
c     Description :
c     -------------
c
c     Cette procedure repere les aretes du multigraphe faisant partie d'un
c     cycle. Le multigraphe est d'ordre NSO, possede NAR aretes definies par
c     des paires de sommets contenues dans BOND.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM2 = 200

      INTEGER  NSO,NAR,BOND(2,NDIM2)
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /ADJACENCE/  ADJL,PALM,NUM
      COMMON  /CYCLE/      ARETE
      COMMON  /PRECEDENT/  PREC

      INTEGER  ADJL(0:4,NDIM1),PALM(2,NDIM2),NUM(NDIM1),PREC(2,NDIM1)
      LOGICAL  ARETE(NDIM1,NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER  I,J,X1,X2,X3
c
c
c     Corps :
c     -------
c
      DO 20 J=1,NSO
         DO 10 I=1,NSO
            ARETE(I,J)=.FALSE.
10       CONTINUE
20    CONTINUE
```

```
c
c...    Nous examinons toutes les aretes du multigraphe :
c
        DO 30 I=1,NAR

           X1=PALM(1,I)
           X2=PALM(2,I)
           IF (NUM(X1).GT.NUM(X2)) THEN
c
c...          Si (X1,X2) est une corde dans le palmier engendre, alors cet arc
c...          engendre un cycle.
c
              ARETE(X1,X2)=.TRUE.
              ARETE(X2,X1)=.TRUE.
c
c...          Nous devons rechercher les aretes faisant partie du cycle engen-
c...          dre par cette corde :
c...          * si le sommet X1 ne fait partie d'aucune chaine precedemment
c...            parcourue, nous "remontons" l'arborescence a partir de ce som-
c...            met X1 jusqu'au sommet X2 en marquant les aretes (non deja mar-
c...            quees) faisant partie de cette chaine.
c
              IF (PREC(1,X1).EQ.0) THEN

                 CALL CHAINE(X2,I)
c
c...          * si le sommet X1 fait partie d'une chaine precedemment parcourue
c...            nous recherchons l'extremite de cette chaine. Soit X1 ce sommet
c...            Si ce nouveau sommet X1 est un descendant de X2, nous devons
c...            encore "remonter" l'arborescence a partir de X1 jusque X2.
c
              ELSE
                 CALL PRECH(X1,X3)
                 IF (NUM(X1).GT.NUM(X2)) CALL CHAINE(X2,PREC(2,X3))
              ENDIF

           ENDIF

30      CONTINUE


        RETURN
        END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      LOGICAL FUNCTION MULLIA
c     ======================
c
c     Description :
c     -------------
c
c     Cette fonction logique est fausse si une liaison autour de laquelle on
c     effectue le fitting flexible (definie par AXE), n'est pas admissible.
c
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NAXE  = 10

      COMMON  /LIAISO/  NBRAXE,AXE

      INTEGER  NBRAXE,AXE(4,NAXE)
c
c     Variables internes :
c     ....................
c
      INTEGER  I,J
      LOGICAL  TEST,LIAD
```

```
c
c
c       Corps :
c       -------
c
        I=0
        TEST=.TRUE.

        DO 10 WHILE (TEST.AND.I.LT.NBRAXE)

          I=I+1
c
c...      Si la fonction logique LIAD est fausse, cela signifie que la liaison
c...      autour de laquelle on effectue le fitting flexible (definie par AXE),
c...      n'est pas admissible i.e. :
c...      - soit il manque une liaison entre les atomes AXE(I) et AXE(I+1),
c...        i=1,3
c...      - soit la liaison est double ou triple,
c...      - soit la liaison est dans un cycle.
c...      Dans ces trois cas, il n'est pas possible d'effectuer un fitting.
c
          TEST=LIAD(AXE(1,I))
10      CONTINUE

        MULLIA=TEST

        RETURN

        END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE DISTAN (NSO,NAR,BOND,DIST)
c     =================
c
c     Description :
c     -------------
c
c     Cette procedure calcule la matrice de distance DIST du multigraphe
c     simple dont NSO est le nombre de sommets, NAR le nombre d'aretes, BOND
c     l'ensemble des paires de sommets formant les aretes.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      INTEGER  NAR,BOND(2,NDIM2),NSO,DIST(NDIM1,NDIM1)
c
c     Variables internes :
c     ....................
c
      PARAMETER  NDIM3 = 250

      INTEGER  I,J,C(NDIM1,4),NUMBRS(NDIM1),IMAX,C11,AT1(NDIM3),INIT,
     1         PREV(NDIM3),INODE,L,CIL,NEXT(NDIM3),AT2(NDIM3),K1,K2,
     2         AT1I,AT2I,M

      LOGICAL  FINISH
c
c
c     Corps :
c     -------
c
      DO 10 I=1,NSO

        DO 20 J=1,4
          C(I,J)=0
20      CONTINUE

        NUMBRS(I)=0

        DO 30 J=1,NAR
          IF (BOND(1,J).EQ.I) THEN
            NUMBRS(I)=NUMBRS(I)+1
            C(I,NUMBRS(I))=BOND(2,J)
          ELSEIF (BOND(2,J).EQ.I) THEN
            NUMBRS(I)=NUMBRS(I)+1
            C(I,NUMBRS(I))=BOND(1,J)
          ENDIF
30      CONTINUE
```

```
          DO 40 J=1,NSO
            DIST(I,J)=0
40        CONTINUE

10      CONTINUE

        IMAX=0
        C11=C(1,1)
        AT1(1)=1
        AT2(1)=C11
        DIST(1,C11)=1
        DIST(C11,1)=1

        INIT=1
        PREV(1)=INIT
        INODE=1

        DO 50 I=1,NSO
          DO 60 L=1,NUMBRS(I)
             CIL=C(I,L)
               IF ((DIST(I,CIL).EQ.0) .AND. (CIL.GT.I)) THEN

                 DIST(I,CIL)=1
                 DIST(CIL,I)=1
                 NEXT(INODE)=INODE+1
                 INODE=INODE+1
                 AT1(INODE)=I
                 AT2(INODE)=CIL
                 PREV(INODE)=INODE-1
               END IF
60        CONTINUE
50      CONTINUE


        NEXT(INODE)=INIT
        PREV(INIT) =INODE


        FINISH=.FALSE.
        I=NEXT(INODE)


        DO 100 WHILE (.not.FINISH)

          K1=0
          K2=0

          AT1I=AT1(I)
          AT2I=AT2(I)
```

```
      DO 110 L=1,NUMBRS(AT1I)
        CIL=C(AT1I,L)

        IF ((DIST(AT2I,CIL).EQ.0) .AND. (AT2I.NE.CIL)) THEN

          K1=K1+1
          DIST(AT2I,CIL)=DIST(AT1I,AT2I) + 1
          DIST(CIL,AT2I)=DIST(AT1I,AT2I) + 1

          IF (K1.EQ.1) THEN

            AT1(I)=AT2(I)
            AT2(I)=CIL
          ELSEIF (K1.GT.1) THEN

            DO 120 M=INODE,I+1,-1
              AT1(M+1)=AT1(M)
              AT2(M+1)=AT2(M)
              PREV(M+1)=PREV(M)+1
              NEXT(M+1)=NEXT(M)+1
120           CONTINUE

            AT1(I+1)=AT2I
            AT2(I+1)=CIL
            PREV(I+1)=I
            NEXT(I+1)=I+2
            PREV(I)=PREV(I)+1
            INODE=INODE+1
            NEXT(INODE)=INIT
            PREV(INIT) =INODE
          END IF
        END IF
110     CONTINUE


      DO 130 L=1,NUMBRS(AT2I)
        CIL=C(AT2I,L)

        IF ((DIST(AT1I,CIL).EQ.0) .AND. (AT1I.NE.CIL)) THEN

          K2=K2+1
          DIST(AT1I,CIL)=DIST(AT2I,AT1I) + 1
          DIST(CIL,AT1I)=DIST(AT2I,AT1I) + 1

          IF ((K1+K2).EQ.1) THEN

            AT2(I)=AT1(I)
            AT1(I)=CIL
          ELSEIF ((K1+K2).GT.1) THEN

            DO 140 M=INODE,I+1,-1
              AT1(M+1)=AT1(M)
              AT2(M+1)=AT2(M)
              PREV(M+1)=PREV(M)+1
              NEXT(M+1)=NEXT(M)+1
140           CONTINUE
```

```
                AT2(I+1)=AT1I
                AT1(I+1)=CIL
                PREV(I+1)=I
                NEXT(I+1)=I+2
                PREV(I)=PREV(I)+1
                INODE=INODE+1
                NEXT(INODE)=INIT
                PREV(INIT) =INODE
              END IF
            END IF
130       CONTINUE

          IF ((K1+K2).EQ.0) THEN

            DO 150 M=I+1,INODE
              AT1(M-1)=AT1(M)
              AT2(M-1)=AT2(M)
              PREV(M-1)=PREV(M)-1
              NEXT(M-1)=NEXT(M)-1
150         CONTINUE
            INODE=INODE-1
            NEXT(INODE)=INIT
            PREV(INIT)=INODE
            IF (I.EQ. (INODE+1) ) I=INIT

            ELSE
              IF ((K1+K2).GT.0) I=NEXT(I+K1+K2-1)

            END IF

            IF(INODE.EQ.0) FINISH=.TRUE.
            IMAX=MAX(INODE,IMAX)

100     CONTINUE

        IF (IMAX.LE.NDIM2) THEN
          RETURN

        ELSE
          WRITE (*,1000) IMAX
                STOP
        END IF

1000    FORMAT(//,' ** WARNING **  In subroutine NODE , IMAX  = ',i4,/,
     *           '     reduce number of atoms and try again !!!'//)

        END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE TORANG (Y,AXE,DANG)
c     ==============================
c
c     Description :
c     -------------
c
c     Calcul de l'angle de torsion DANG, i.e. de l'angle forme par le plan
c     determine par les 'points' Y(axe(2),*),Y(axe(3),*),Y(axe(4),*) et par
c     le plan determine par les 'points' Y(axe(1),*),Y(axe(2),*),Y(axe(3),*).
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100

      INTEGER           AXE(4)
      DOUBLE PRECISION  Y(NDIM1,3),DANG
c
c     Variables internes :
c     ....................
c
      INTEGER           I,SIGNE
      DOUBLE PRECISION  A,B,C,D,M(3),N(3),O(3),Y1(3),Y2(3)
c
c
c     Corps :
c     -------
c
c     1. Evaluation de la valeur absolue de l'angle de torsion.
c
c     1.1. Calcul du vecteur M, normal au plan determine par les 'points'
c          Y(axe(1),*),Y(axe(2),*),Y(axe(3),*).
c
      DO 10 I=1,3
         Y1(I)=Y(AXE(3),I)-Y(AXE(1),I)
         Y2(I)=Y(AXE(2),I)-Y(AXE(1),I)
10    CONTINUE
      CALL PRODVER(Y1,Y2,A,B,C)
      D=SQRT(A**2+B**2+C**2)
      M(1)=A/D
      M(2)=B/D
      M(3)=C/D
```

```
c
c       1.2. Calcul du vecteur N, normal au plan determine par les 'points'
c            Y(axe(2),*),Y(axe(3),*),Y(axe(4),*).
c
        DO 20 I=1,3
           Y1(I)=Y(AXE(3),I)-Y(AXE(4),I)
           Y2(I)=Y(AXE(2),I)-Y(AXE(4),I)
20      CONTINUE
        CALL PRODVER(Y1,Y2,A,B,C)
        D=SQRT(A**2+B**2+C**2)
        N(1)=A/D
        N(2)=B/D
        N(3)=C/D
c
c       1.3. Calcul de la valeur absolue de l'angle de torsion.
c
        DANG=ACOS(N(1)*M(1)+N(2)*M(2)+N(3)*M(3))
c
c       2. Determination du signe de l'angle.
c
        CALL PRODVER(N,M,O(1),O(2),O(3))
        I=1
        SIGNE=0
        DO 30 WHILE(SIGNE.EQ.0.AND.I.LE.3)
           IF ((Y(AXE(2),I)-Y(AXE(3),I))*O(I).LT.0.DO) THEN
              SIGNE=-1
           ELSEIF ((Y(AXE(2),I)-Y(AXE(3),I))*O(I).GT.0.DO) THEN
              SIGNE=1
           ENDIF
           I=I+1
30      CONTINUE
        IF (SIGNE.EQ.-1) DANG=-DANG


        RETURN
        END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE PRODVER (X,Y,A,B,C)
c     ==================
c
c     Description :
c     -------------
c
c     Cette procedure calcule le produit vectoriel des vecteurs X et Y.
c     Le vecteur ainsi obtenu est le vecteur (A,B,C)t.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  X(3),Y(3),A,B,C
c
c
c     Corps :
c     -------
c
      A=X(2)*Y(3)-X(3)*Y(2)
      B=X(3)*Y(1)-X(1)*Y(3)
      C=X(1)*Y(2)-X(2)*Y(1)


      RETURN
      END
c
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE COSDIR (MAT,AXE,CD)
c     ==================
c
c     Description :
c     -------------
c
c     Cette procedure calcule les cosinus directeurs CD(1), CD(2), CD(3)
c     de l'axe de rotation defini par les atomes AXE(2) et AXE(3) de la
c     molecule MAT.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100

      INTEGER           AXE(4)
      DOUBLE PRECISION  MAT(NDIM1,3),CD(3)
c
c     Variables internes :
c     ....................
c
      DOUBLE PRECISION  B(3),D
c
c
c     Corps :
c     .......
c
      B(1)=MAT(AXE(3),1)-MAT(AXE(2),1)
      B(2)=MAT(AXE(3),2)-MAT(AXE(2),2)
      B(3)=MAT(AXE(3),3)-MAT(AXE(2),3)

      D=SQRT(B(1)*B(1)+B(2)*B(2)+B(3)*B(3))

      CD(1)=B(1)/D
      CD(2)=B(2)/D
      CD(3)=B(3)/D

      RETURN
      END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE TRANSF (MAT,AXE,T)
      =================
c
c     Description :
c     -------------
c
c     Cette procedure calcule la translation T permettant de faire passer l'
c     axe (AXE) par l'origine du systeme de reference de la molecule mobile.
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100

      INTEGER           AXE(4)
      DOUBLE PRECISION  MAT(NDIM1,3),T(3)
c
c
c     Corps :
c     -------
c
      T(1)=MAT(AXE(3),1)
      T(2)=MAT(AXE(3),2)
      T(3)=MAT(AXE(3),3)

      RETURN
      END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE TRANS (MAT,NAT,T)
      ================
c
c
c     Description :
c     -------------
c
c     Cette procedure translate d'un vecteur T la molecule MAT comprenant
c     NAT atomes.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100

      INTEGER           NAT
      DOUBLE PRECISION  MAT(NDIM1,3),T(3)
c
c     Variables internes :
c     ....................
c
      INTEGER           I
c
c
c     Corps :
c     -------
c
       DO 10 I=1,NAT
         MAT(I,1)=MAT(I,1)-T(1)
         MAT(I,2)=MAT(I,2)-T(2)
         MAT(I,3)=MAT(I,3)-T(3)
10     CONTINUE

      RETURN
      END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE MOBAT
c     ================
c
c     Description :
c     -------------
c
c     Cette procedure (1) cree ATOMOB, l'ensemble des atomes  de la molecule
c     mobile qui pivotent lors de la rotation autour de chaque axe et
c     (2) construit AME les atomes a egaler se trouvant dans chaque partie
c     mobile de la molecule.
c
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NAXE  = 10

      COMMON  /MOBILE/  YAT
      COMMON  /LIAISO/  NBRAXE,AXE,ATOMOB
      COMMON  /DISTA/   DIST
      COMMON  /DIVISI/  NBATEG,AME

      INTEGER   YAT,NBATEG,AME(0:NDIM1,0:NAXE),
     1          NBRAXE,AXE(4,NAXE),ATOMOB(0:NDIM1,NAXE),
     2          DIST(NDIM1,NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER   I,J,DJ,DK
      LOGICAL   APRT
c
c
c     Corps :
c     -------
c
      DO 10 I=1,NBRAXE
c
c...     (1) Determination de la partie mobile de la molecule mobile par
c...     rapport a l'axe I.
c
         ATOMOB(0,I)=0
```

```
c
c...        Pour chaque atome J de Y (sauf l'atome AXE(3)) : si la longueur du
c...        plus court chemin de cet atome a l'atome AXE(2) est superieure a
c...        celle de ce meme atome a l'atome AXE(3), alors l'atome bougera lors
c...        de la rotation.
c
            DO 20 J=1,YAT

               IF (J.NE.AXE(3,I)) THEN
                  DJ=DIST(AXE(2,I),J)
                  DK=DIST(AXE(3,I),J)
                  IF ((DJ-DK).EQ.1) THEN
c
c...                 ATOMOB(0,I) est le nombre d'atomes qui bougent lors de
c...                 la rotation.
c
                     ATOMOB(0,I)=ATOMOB(0,I)+1
                     ATOMOB(ATOMOB(0,I),I)=J
                  ENDIF
               ENDIF

20          CONTINUE

10       CONTINUE
c
c...     (2) Determination des atomes a egaler se trouvant dans chaque partie
c...     de la molecule mobile.
c
         DO 30 I=1,NBRAXE

            AME(0,I)=0

            DO 40 J=1,NBATEG
               IF (APRT(AME(J,0),ATOMOB(0,I))) THEN
c
c...              AME(0,I) est le nombre d'atomes a egaler dans la ieme partie.
c
                  AME(0,I)=AME(0,I)+1
                  AME(AME(0,I),I)=AME(J,0)
               ENDIF

40          CONTINUE

30       CONTINUE

         RETURN
         END
c
c -----------------------------------------------------------------------
c -----------------------------------------------------------------------
```

```
c-----------------------------------------------------------------------
c-----------------------------------------------------------------------
c
      SUBROUTINE MOBATBIS
c     ===================
c
c     Description :
c     -------------
c
c     Cette procedure cree ATOMOB, l'ensemble des atomes qui bougent lors
c     de la rotation de la molecule mobile Y autour de l'axe defini par
c     les atomes AXE(2) et AXE(3).
c
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /MOBILE/  YAT,Y
      COMMON  /LIAISO/  AXE,ATOMOB
      COMMON  /DISTA/   DIST

      INTEGER           YAT,AXE(4),DIST(NDIM1,NDIM1),ATOMOB(0:NDIM1)
      DOUBLE PRECISION  Y(NDIM1,3)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,DJ,DK

c
c
c     Corps :
c     -------
c
      ATOMOB(0)=0
c
c...  Pour chaque atome I de Y (sauf l'atome AXE(3)) : si la "distance" de
c...  cet atome a l'atome AXE(2) est superieure a la "distance" de cet atome
c...  a l'atome AXE(3), i.e. si le nombre minimal de liaisons (d'aretes)
c...  entre cet atome et AXE(2) est superieur au nombre minimal de liaisons
c...  entre cet atome et AXE(3), alors l'atome bougera lors de la rotation.
c
      DO 10 I=1,YAT

         IF (I.NE.AXE(3)) THEN
            DJ=DIST(AXE(2),I)
            DK=DIST(AXE(3),I)
            IF ((DJ-DK).eq.1) THEN
```

```
c
c...              ATOMOB(0) est le nombre d'atomes qui bougent lors de
c...              la rotation.
c
                  ATOMOB(0)=ATOMOB(0)+1
                  ATOMOB(ATOMOB(0))=I
               ENDIF
            ENDIF

10       CONTINUE


         RETURN
         END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE ROTATI (delta)
      =================
c
c
c     Description :
c     -------------
c
c     Cette procedure effectue la rotation d'un angle DELTA(i) de la
c     partie mobile de la molecule Y autour de chaque axe i.
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      PARAMETER  NAXE = 10

      DOUBLE PRECISION  DELTA(NAXE)
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /MOBILE/  YAT,Y
      COMMON  /LIAISO/  NBRAXE,AXE,ATOMOB

      INTEGER           YAT,NBRAXE,AXE(4,NAXE),
     1                  ATOMOB(0:NDIM1,NAXE)
      DOUBLE PRECISION  Y(NDIM1,3)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
      DOUBLE PRECISION  CD(3),T(3),
     1                  ROT(3,3),B(3)
c
c
c     Corps :
c     -------
c
      DO 10 I=1,NBRAXE
c
c...    Ces routines calculent les cosinus directeurs du Ieme axe,
c...    ainsi que la translation deplacant l'axe a l'origine.
c
        CALL COSDIR(Y,AXE(1,I),CD)
        CALL TRANSF(Y,AXE(1,I),T)
```

```
c
c...      Translation de la partie mobile de la molecule pour l'axe considere.
c
          DO 20 J=1,ATOMOB(0,I)
            Y(ATOMOB(J,I),1)=Y(ATOMOB(J,I),1)-T(1)
            Y(ATOMOB(J,I),2)=Y(ATOMOB(J,I),2)-T(2)
            Y(ATOMOB(J,I),3)=Y(ATOMOB(J,I),3)-T(3)
20        CONTINUE
c
c...      La procedure MATROT retourne la matrice de rotation (d'un angle
c...      DELTA(i)) autour d'un axe dont les cosinus directeurs sont CD(1),
c...      CD(2), CD(3)).
c
          CALL MATROT(DELTA(I),CD,ROT)
c
c...      On effectue la rotation des atomes (seuls les atomes contenus dans
c...      ATOMOB subissent l'effet de la rotation).
c
          DO 30 J=1,ATOMOB(0,I)
           B(1)=ROT(1,1)*Y(ATOMOB(J,I),1)+ROT(1,2)*Y(ATOMOB(J,I),2)
     +          +ROT(1,3)*Y(ATOMOB(J,I),3)
           B(2)=ROT(2,1)*Y(ATOMOB(J,I),1)+ROT(2,2)*Y(ATOMOB(J,I),2)
     +          +ROT(2,3)*Y(ATOMOB(J,I),3)
           B(3)=ROT(3,1)*Y(ATOMOB(J,I),1)+ROT(3,2)*Y(ATOMOB(J,I),2)
     +          +ROT(3,3)*Y(ATOMOB(J,I),3)
           Y(ATOMOB(J,I),1)=B(1)
           Y(ATOMOB(J,I),2)=B(2)
           Y(ATOMOB(J,I),3)=B(3)
30        CONTINUE
c
c...       Translation inverse de la partie mobile de la molecule.
c
          DO 40 J=1,ATOMOB(0,I)
             Y(ATOMOB(J,I),1)=Y(ATOMOB(J,I),1)+T(1)
             Y(ATOMOB(J,I),2)=Y(ATOMOB(J,I),2)+T(2)
             Y(ATOMOB(J,I),3)=Y(ATOMOB(J,I),3)+T(3)
40        CONTINUE
10       CONTINUE

         RETURN
         END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE ROTATIBIS (Z,DELTA)
      ==========
c
c
c     Description :
c     -------------
c
c     Cette procedure effectue la rotation d'un angle DELTA de la molecule Z,
c     rotation suivant l'axe dont les cosinus directeurs sont CD(1), CD(2) et
c     CD(3).
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      PARAMETER  NDIM1 = 100

      DOUBLE PRECISION  Z(NDIM1,3),DELTA
c
c     Commons :
c     .........
c
      COMMON  /LIAISO/  AXE,ATOMOB
      COMMON  /DISTA/   DIST
      COMMON  /TRANF/   CD,T

      INTEGER           ATOMOB(0:NDIM1),AXE(4),DIST(NDIM1,NDIM1)
      DOUBLE PRECISION  CD(3),T(3)
c
c     Variables internes :
c     ....................
c
      INTEGER           I
      DOUBLE PRECISION  ROT(3,3),AUX(3)
c
c
c     Corps :
c     -------
c
c...  La procedure MATROT retourne le matrice de rotation (d'un angle DELTA
c...  autour d'un axe dont les cosinus directeurs sont CD(1), CD(2), CD(3)).
c
      CALL MATROT(DELTA,CD,ROT)
```

```
c
c...    On effectue la rotation des atomes (seuls les atomes contenus dans
c...    ATOMOB subissent l'effet de la rotation).
c
        DO 10 I=1,ATOMOB(0)

           AUX(1)=ROT(1,1)*Z(ATOMOB(I),1)+ROT(1,2)*Z(ATOMOB(I),2)
     +           +ROT(1,3)*Z(ATOMOB(I),3)
           AUX(2)=ROT(2,1)*Z(ATOMOB(I),1)+ROT(2,2)*Z(ATOMOB(I),2)
     +           +ROT(2,3)*Z(ATOMOB(I),3)
           AUX(3)=ROT(3,1)*Z(ATOMOB(I),1)+ROT(3,2)*Z(ATOMOB(I),2)
     +           +ROT(3,3)*Z(ATOMOB(I),3)

           Z(ATOMOB(I),1)=AUX(1)
           Z(ATOMOB(I),2)=AUX(2)
           Z(ATOMOB(I),3)=AUX(3)

10      CONTINUE


        RETURN
        END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE MATROT (DELTA,CD,ROT)
      ================
c
c
c     Description :
c     -------------
c
c     Cette procedure calcule la matrice de rotation ROT, rotation d'un angle
c     DELTA autour d'un axe dont les cosinus directeurs sont CD(1), CD(2) et
c     CD(3).
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  ROT(3,3),CD(3),DELTA
c
c
c     Corps :
c     -------
c
      ROT(1,1)=COS(delta)+(1-COS(delta))*CD(1)*CD(1)
      ROT(2,2)=COS(delta)+(1-COS(delta))*CD(2)*CD(2)
      ROT(3,3)=COS(delta)+(1-COS(delta))*CD(3)*CD(3)
      ROT(1,2)=(1-COS(delta))*CD(1)*CD(2)-SIN(delta)*CD(3)
      ROT(1,3)=(1-COS(delta))*CD(1)*CD(3)+SIN(delta)*CD(2)
      ROT(2,1)=(1-COS(delta))*CD(2)*CD(1)+SIN(delta)*CD(3)
      ROT(2,3)=(1-COS(delta))*CD(2)*CD(3)-SIN(delta)*CD(1)
      ROT(3,1)=(1-COS(delta))*CD(3)*CD(1)-SIN(delta)*CD(2)
      ROT(3,2)=(1-COS(delta))*CD(3)*CD(2)+SIN(delta)*CD(1)


      RETURN
      END
c
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      LOGICAL FUNCTION APRT (IND,ATOMOB)
      ==========
c
c
c     Cette fonction teste si l'atome IND appartient a la partie de la
c     molecule definie par ATOMOB.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      PARAMETER  NDIM1 = 100

      INTEGER  IND,ATOMOB(0:NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER  I
c
c
c     Corps :
c     -------
c
      APRT=.FALSE.
      I=0
      DO 10 WHILE (.NOT.APRT.AND.(I.LT.ATOMOB(0)))
         I=I+1
         APRT=IND.EQ.ATOMOB(I)
10    CONTINUE


      RETURN
      END
c
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE RMS (NORM,FCT,FNORM)
c     ===============
c
c     Cette routine calcule la distance entre chaque atome a egaler NORM,
c     la somme des distances au carre FCT et le rms FNORM.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      PARAMETER  NDIM1 = 100

      DOUBLE PRECISION  NORM(NDIM1),FCT,FNORM
c
c     Commons :
c     .........
c
      PARAMETER  NAXE = 10

      COMMON  /FIXE/     XAT,X
      COMMON  /MOBILE/   YAT,Y
      COMMON  /DIVISI/   NBATEG,AME,AFE

      INTEGER            XAT,YAT,NBATEG,AME(0:NDIM1,0:NAXE),AFE(NDIM1)
      DOUBLE PRECISION   X(NDIM1,3),Y(NDIM1,3)
c
c     Variables internes :
c     ....................
c
      INTEGER            I
c
c
c     Corps :
c     -------
c
c
      FCT=0.0
      DO 10 I=1,NBATEG
c
c...    Calcul de la distance entre les Ieme atomes a egaler.
c
        NORM(I)=(X(AFE(I),1)-Y(AME(I,0),1))**2
     +          +(X(AFE(I),2)-Y(AME(I,0),2))**2
     +          +(X(AFE(I),3)-Y(AME(I,0),3))**2
c
c...    Calcul de la somme des distances.
c
        FCT=FCT+NORM(I)
        NORM(I)=SQRT(NORM(I))
10    CONTINUE
```

```
c
c...     Calcul du rms.
c
         FNORM=SQRT(FCT/NBATEG)


         RETURN
         END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE RMSBIS (NORM,FCT,FNORM)
      ==================
c
c     Cette routine calcule la distance entre chaque atome a egaler NORM,
c     la somme des distances au carre FCT et le rms FNORM.
c
c
c     Declarations :
c     --------------
c
c     Parametre :
c     ...........
c
      PARAMETER  NDIM1 = 100

      DOUBLE PRECISION  NORM(NDIM1),FCT,FNORM
c
c     Commons :
c     .........
c
      COMMON  /FIXE/     XAT,X
      COMMON  /MOBILE/   YAT,Y
      COMMON  /DIVISI/   NBATEG,AME,AFE

      INTEGER            XAT,YAT,NBATEG,AME(NDIM1),AFE(NDIM1)
      DOUBLE PRECISION   X(NDIM1,3),Y(NDIM1,3)
c
c     Variables internes :
c     ....................
c
      INTEGER            I
c
c
c     Corps :
c     -------
c
c
      FCT=0.0
      DO 10 I=1,NBATEG
c
c...    Calcul de la distance entre les Ieme atomes a egaler.
c
        NORM(I)=(X(AFE(I),1)-Y(AME(I),1))**2
     +          +(X(AFE(I),2)-Y(AME(I),2))**2
     +          +(X(AFE(I),3)-Y(AME(I),3))**2
c
c...    Calcul de la somme des distances.
c
        FCT=FCT+NORM(I)
        NORM(I)=SQRT(NORM(I))
10     CONTINUE
```

```
c
c...    Calcul du rms.
c
        FNORM=SQRT(FCT/NBATEG)


        RETURN
        END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      DOUBLE PRECISION FUNCTION DEG (DELTA)
c     =============================
c
c     Description :
c     -------------
c
c     Cette fonction convertit en degre l'angle DELTA donne en radian.
c
c
c     Declaration :
c     -------------
c
c     Parametre :
c     ...........
c
      DOUBLE PRECISION DELTA
c
c
c     Corps :
c     -----
c
      DEG = DELTA*180/ACOS(-1.DO)


      RETURN
      END
c
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      DOUBLE PRECISION FUNCTION RAD (DELTA)
c     ============================
c
c     Description :
c     -------------
c
c     Cette fonction convertit en radian l'angle DELTA donne en degre.
c
c
c     Declaration :
c     -------------
c
c     Parametre :
c     ...........
c
      DOUBLE PRECISION DELTA
c
c
c     Corps :
c     -------
c
      RAD = DELTA*ACOS(-1.DO)/180


      RETURN
      END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE INFCT (IDP,CCS,CSN)
c     ==============
c
c     Description :
c     -------------
c
c     Cette procedure calcule les constantes de la fonction objectif.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  IDP,CCS,CSN
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /FIXE/     XAT,X
      COMMON  /MOBILE/   YAT,Y
      COMMON  /DIVISI/   NBATEG,AME,AFE
      COMMON  /TRANF/    CD

      INTEGER            XAT,YAT,NBATEG,AME(NDIM1),AFE(NDIM1)
      DOUBLE PRECISION   X(NDIM1,3),Y(NDIM1,3),CD(3)
c
c     Variables internes :
c     ....................
c
      INTEGER            I
      DOUBLE PRECISION   COORD,DCS,CCS1,CCS2
c
c
c     Corps :
c     -------
c
c     1. Calcul des termes independants.
c
      COORD=0
      DO 10 I=1,NBATEG
         COORD=COORD+Y(AME(I),1)*Y(AME(I),1)+Y(AME(I),2)
     +               *Y(AME(I),2)+Y(AME(I),3)*Y(AME(I),3)
     +               +X(AFE(I),1)*X(AFE(I),1)+X(AFE(I),2)
     +               *X(AFE(I),2)+X(AFE(I),3)*X(AFE(I),3)
10    CONTINUE

      DCS=0
      DO 20 I=1,NBATEG
         DCS=DCS+(X(AFE(I),1)*CD(1)+X(AFE(I),2)*CD(2)
     +          + X(AFE(I),3)*CD(3))*(Y(AME(I),1)*CD(1)
     +          + Y(AME(I),2)*CD(2)+Y(AME(I),3)*CD(3))
20    CONTINUE
```

```
      IDP=COORD-2*DCS
C
C     2. Calcul des coefficients en cosinus.
C
      CCS1=0
      DO 30 I=1,NBATEG
         CCS1=CCS1+(CD(1)*CD(1)-1)*Y(AME(I),1)*X(AFE(I),1)
     +             +(CD(2)*CD(2)-1)*Y(AME(I),2)*X(AFE(I),2)
     +             +(CD(3)*CD(3)-1)*Y(AME(I),3)*X(AFE(I),3)
30    CONTINUE

      CCS2=0
      DO 40 I=1,NBATEG
        CCS2=CCS2+CD(1)*CD(2)*(Y(AME(I),2)*X(AFE(I),1)
     +             +Y(AME(I),1)*X(AFE(I),2))
     +             +CD(2)*CD(3)*(Y(AME(I),3)*X(AFE(I),2)
     +             +Y(AME(I),2)*X(AFE(I),3))
     +             +CD(3)*CD(1)*(Y(AME(I),1)*X(AFE(I),3)
     +             +Y(AME(I),3)*X(AFE(I),1))
40    CONTINUE

      CCS=2*(CCS1+CCS2)
C
C     3. Calcul des coefficients en sinus.
C
      CSN=0
      DO 50 I=1,NBATEG
         CSN=CSN+CD(3)*(Y(AME(I),2)*X(AFE(I),1)
     +            -Y(AME(I),1)*X(AFE(I),2))
     +            +CD(2)*(Y(AME(I),1)*X(AFE(I),3)
     +            -Y(AME(I),3)*X(AFE(I),1))
     +            +CD(1)*(Y(AME(I),3)*X(AFE(I),2)
     +            -Y(AME(I),2)*X(AFE(I),3))
50    CONTINUE

      CSN=2*CSN

      RETURN
      END
C
C ---------------------------------------------------------------------------
C ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE CONT (DELTA,ETOR,EVDW,ECOUL)
c     ===============
c
c     Description :
c     -------------
c
c     Cette procedure calcule les energies de torsion, de Van Der Waals et
c     de Coulomb de la molecule mobile Y qui a subi une rotation DELTA.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  DELTA,ETOR,EVDW,ECOUL
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /MOBILE/      YAT,Y
      COMMON  /LIAISO/      AXE,ATOMOB
      COMMON  /DISTA/       DIST
      COMMON  /ENERGI/      EPS,RVDW,QC,VT,ST,NT
      COMMON  /TRANF/       CD
      COMMON  /ADJACENCE/   ADJL

      INTEGER           YAT,ATOMOB(0:NDIM1),AXE(4),DIST(NDIM1,NDIM1),
     1                  ST(9),ADJL(0:4,NDIM1)
      DOUBLE PRECISION  Y(NDIM1,3),CD(3),
     1                  EPS(NDIM1,NDIM1),RVDW(NDIM1),QC(NDIM1),VT(9),
     2                  NT(9)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J,L,M,N,AX(4)
      DOUBLE PRECISION  Z(NDIM1,3),DANG,RACAR
c
c
c     Corps :
c     -------
c
c     1. Rotation de la molecule Y.
c
      DO 20 J=1,YAT
         DO 10 I=1,3
            Z(J,I)=Y(J,I)
10       CONTINUE
20    CONTINUE
```

```
      CALL ROTATIBIS(Z,DELTA)
C
C     2. Calcul des energies.
C
C     2.1. Calcul de l'energie de torsion.
C
      AX(2)=AXE(2)
      AX(3)=AXE(3)
      ETOR=0.0
      L=1
      DO 26 M=1,ADJL(0,AXE(2))
         IF (ADJL(M,AXE(2)).NE.AXE(3)) THEN
            DO 24 N=1,ADJL(0,AXE(3))
               IF (ADJL(N,AXE(3)).NE.AXE(2)) THEN
C
C...              Calcul de l'angle de torsion
C
                  AX(1)=ADJL(M,AXE(2))
                  AX(4)=ADJL(N,AXE(3))
                  CALL TORANG(Z,AX,DANG)

                  ETOR=VT(L)*(1+ST(L)*COS(NT(L)*DANG))

                  L=L+1

               ENDIF
24          CONTINUE
         ENDIF
26    CONTINUE
C
C     2.2 Calcul des energies de Coulomb et de Van Der Waals.
C
      ECOUL=0.D0
      EVDW=0.D0

      DO 50 I=1,YAT
         DO 40 J=I+1,YAT
C
C...        Calcul du carre de la distance entre l'atome Z(I,.) et
C...        l'atome Z(J,.).
C
            RACAR=(Z(I,1)-Z(J,1))**2+
     1            (Z(I,2)-Z(J,2))**2+
     2            (Z(I,3)-Z(J,3))**2
```

```
c
c...          Si les atomes Z(I,.) et Z(J,.) sont au moins 1,4 et si ils
c...          sont suffisamment proches, on calcule leurs energies.
c
              IF ((DIST(I,J).GE.3).AND.((SQRT(RACAR)).LE.9.DO)) THEN
                 ECOUL=ECOUL+QC(I)*QC(J)*332.17/(SQRT(RACAR))
                 EVDW=EVDW+EPS(I,J)*(-2.25/(RACAR**3)
     +                *((RVDW(I)+RVDW(J))**6)+8.28D5
     +                *EXP(-1/0.0736*SQRT(RACAR)/(RVDW(I)+RVDW(J))))
              ENDIF

40          CONTINUE
50        CONTINUE


          RETURN
          END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE CONT1 (DELTA,ETOR1,EVDW1,ECOUL1)
      ===================
c
c
c     Description :
c     -------------
c
c     Cette procedure calcule  la valeur de la derivee premiere des energies
c     de torsion, de Van Der Waals et de Coulomb de la molecule mobile Y
c     qui a subi une rotation DELTA.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  DELTA,ETOR1,EVDW1,ECOUL1
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /MOBILE/      YAT,Y
      COMMON  /LIAISO/      AXE,ATOMOB
      COMMON  /DISTA/       DIST
      COMMON  /ENERGI/      EPS,RVDW,QC,VT,ST,NT
      COMMON  /TRANF/       CD
      COMMON  /ADJACENCE/   ADJL

      INTEGER           YAT,ATOMOB(0:NDIM1),AXE(4),DIST(NDIM1,NDIM1),
     1                  ST(9),ADJL(0:4,NDIM1)
      DOUBLE PRECISION  Y(NDIM1,3),CD(3),
     1                  EPS(NDIM1,NDIM1),RVDW(NDIM1),QC(NDIM1),
     2                  VT(9),NT(9)

c
c     Variables internes :
c     ....................
c
      INTEGER           I,J,L,M,N,AX(4)
      DOUBLE PRECISION  Z(NDIM1,3),
     1                  DANG,RACAR,DERA1,CCS1,CCS2,CSN
      LOGICAL           APRT
c
c
c     Corps :
c     -------
c
c     1. Rotation de la molecule Y.
c
      DO 20 J=1,YAT
         DO 10 I=1,3
            Z(J,I)=Y(J,I)
10       CONTINUE
20    CONTINUE
```

```
      CALL ROTATIBIS(Z,DELTA)
C
C     2. Calcul de la derivee premiere des energies.
C
C     2.1. Calcul de la derivee de l'energie de torsion.
C
      AX(2)=AXE(2)
      AX(3)=AXE(3)
      ETOR1=0.0
      L=1
      DO 26 M=1,ADJL(0,AXE(2))
         IF (ADJL(M,AXE(2)).NE.AXE(3)) THEN
            DO 24 N=1,ADJL(0,AXE(3))
               IF (ADJL(N,AXE(3)).NE.AXE(2)) THEN
C
C...              Calcul de l'angle de torsion
C
                  AX(1)=ADJL(M,AXE(2))
                  AX(4)=ADJL(N,AXE(3))
                  CALL TORANG(Z,AX,DANG)

                  ETOR1=-VT(L)*ST(L)*NT(L)*SIN(NT(L)*DANG)

                  L=L+1

               ENDIF
24          CONTINUE
         ENDIF
26    CONTINUE
C
C...  Calcul de la derivee des energies de Coulomb et de Van Der Waals.
C
      ECOUL1=0.D0
      EVDW1=0.D0

      DO 50 I=1,YAT
         DO 40 J=I+1,YAT
C
C...        Calcul du carre de la distance entre l'atome Z(I,.) et
C...        l'atome Z(J,.).
C
            RACAR=(Z(I,1)-Z(J,1))**2+
     1            (Z(I,2)-Z(J,2))**2+
     2            (Z(I,3)-Z(J,3))**2
```

```
C
C...          Si Z(I,.) a bouge lors de la rotation et Z(J,.) est reste fixe,
C...          ou si Z(I,.) est reste fixe lors de la rotation et Z(J,.)
C...          a bouge, on calcule la derivee de la fonction "carre de la dis-
C...          tance entre Z(I,.) etZ(J,.)". Sinon la derivee est nulle,
C...          la distance entre Z(I,.) et Z(J,.) restant inchangee lors de
C...          la rotation.
C
              IF ((APRT(I,ATOMOB).AND..NOT.APRT(J,ATOMOB)).OR.
             1    (.NOT.APRT(I,ATOMOB).AND.APRT(J,ATOMOB))) THEN

                 CCS1=(CD(1)*CD(1)-1)*Y(J,1)*Y(I,1)
             +       +(CD(2)*CD(2)-1)*Y(J,2)*Y(I,2)
             +       +(CD(3)*CD(3)-1)*Y(J,3)*Y(I,3)

                 CCS2=CD(1)*CD(2)*(Y(J,2)*Y(I,1)+Y(J,1)*Y(I,2))
             +       +CD(2)*CD(3)*(Y(J,3)*Y(I,2)+Y(J,2)*Y(I,3))
             +       +CD(3)*CD(1)*(Y(J,1)*Y(I,3)+Y(J,3)*Y(I,1))

                 CSN=CD(3)*(Y(J,2)*Y(I,1)-Y(J,1)*Y(I,2))
             +      +CD(2)*(Y(J,1)*Y(I,3)-Y(J,3)*Y(I,1))
             +      +CD(1)*(Y(J,3)*Y(I,2)-Y(J,2)*Y(I,3))

                 DERA1=-2*SIN(DELTA)*(CCS1+CCS2)+2*COS(DELTA)*CSN
              ELSE
                 DERA1=0.D0
              ENDIF
C
C...          Si les atomes Z(I,.) et Z(J,.) sont au moins 1,4 et si ils sont
C...          suffisamment proches, on calcule la derivee de leurs energies.
C
              IF ((DIST(I,J).GE.3).AND.((SQRT(RACAR)).LE.9.D0)) THEN
                 ECOUL1=ECOUL1-QC(I)*QC(J)*0.5*332.17*DERA1
             +                 /(SQRT(RACAR)*RACAR)

                 EVDW1=EVDW1+EPS(I,J)*(3*2.25*((RVDW(I)+RVDW(J))**6)
             +                 *DERA1/(RACAR**4)+8.28D5*0.5*(-1/0.0736)
             +                 *EXP(-1/0.0736*SQRT(RACAR)/(RVDW(I)+RVDW(J)))
             +                 /(SQRT(RACAR)*(RVDW(I)+RVDW(J)))*DERA1)
               ENDIF

40          CONTINUE
50        CONTINUE


          RETURN
          END
C
C -------------------------------------------------------------------------
C -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE CONT2 (DELTA,ETOR2,EVDW2,ECOUL2)
c     ===================
c
c     Description :
c     -------------
c
c     Cette procedure calcule la derivee seconde des energies de torsion,
c     de Van Der Waals et de Coulomb de la molecule mobile Y qui a subi
c     une rotation DELTA.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      DOUBLE PRECISION  DELTA,ETOR2,EVDW2,ECOUL2
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /MOBILE/      YAT,Y
      COMMON  /LIAISO/      AXE,ATOMOB
      COMMON  /DISTA/       DIST
      COMMON  /ENERGI/      EPS,RVDW,QC,VT,ST,NT
      COMMON  /TRANF/       CD
      COMMON  /ADJACENCE/   ADJL

      INTEGER           YAT,ATOMOB(0:NDIM1),AXE(4),DIST(NDIM1,NDIM1),
     1                  ST(9),ADJL(0:4,NDIM1)
      DOUBLE PRECISION  Y(NDIM1,3),CD(3),
     1                  EPS(NDIM1,NDIM1),RVDW(NDIM1),QC(NDIM1),
     2                  VT(9),NT(9)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J,L,M,N,AX(4)
      DOUBLE PRECISION  Z(NDIM1,3),
     1                  DANG,RACAR,DERA1,DERA2,CCS1,CCS2,CSN,CTS
      LOGICAL           APRT
c
c
c     Corps :
c     -------
c
c     1. Rotation de la molecule Y.
c
      DO 20 J=1,YAT
         DO 10 I=1,3
            Z(J,I)=Y(J,I)
10       CONTINUE
20    CONTINUE
```

```
      CALL ROTATIBIS(Z,DELTA)
C
C     2. Calcul de la derivee seconde des energies.
C
C     2.1. Calcul de la derivee seconde de l'energie de torsion.
C
      AX(2)=AXE(2)
      AX(3)=AXE(3)
      ETOR2=0.0
      L=1
      DO 26 M=1,ADJL(0,AXE(2))
         IF (ADJL(M,AXE(2)).NE.AXE(3)) THEN
            DO 24 N=1,ADJL(0,AXE(3))
               IF (ADJL(N,AXE(3)).NE.AXE(2)) THEN
C
C...              Calcul de l'angle de torsion
C
                  AX(1)=ADJL(M,AXE(2))
                  AX(4)=ADJL(N,AXE(3))
                  CALL TORANG(Z,AX,DANG)

                  ETOR2=-VT(L)*ST(L)*NT(L)*NT(L)*COS(NT(L)*DANG)

                  L=L+1

               ENDIF
24          CONTINUE
         ENDIF
26    CONTINUE
C
C     2.2 Calcul de la derivee seconde des energies de Coulomb et de Van
C         Der Waals.
C
      ECOUL2=0.D0
      EVDW2=0.D0

      DO 50 I=1,YAT
         DO 40 J=I+1,YAT
C
C...        Calcul du carre de la distance entre l'atome Z(I,.) et
C...        l'atome Z(J,.).
C
            RACAR=(Z(I,1)-Z(J,1))**2+
     1            (Z(I,2)-Z(J,2))**2+
     2            (Z(I,3)-Z(J,3))**2
```

```
C
C...        Si Z(I,.) a bouge lors de la rotation et Z(J,.) est reste fixe,
C...        ou si Z(I,.) est reste fixe lors de la rotation et Z(J,.)
C...        a bouge, on calcule les derivees premiere et seconde de la fonc-
C...        tion "carre de la distance entre Z(I,.) etZ(J,.)". Sinon la deri-
C...        vee premiere est nulle (ainsi que la derivee seconde !),
C...        la distance entre Z(I,.) et Z(J,.) restant inchangee lors de
C...        la rotation.
C
            IF ((APRT(I,ATOMOB).AND..NOT.APRT(J,ATOMOB)).OR.
           1   (.NOT.APRT(I,ATOMOB).AND.APRT(J,ATOMOB))) THEN

               CCS1=(CD(1)*CD(1)-1)*Y(J,1)*Y(I,1)
           +       +(CD(2)*CD(2)-1)*Y(J,2)*Y(I,2)
           +       +(CD(3)*CD(3)-1)*Y(J,3)*Y(I,3)

               CCS2=CD(1)*CD(2)*(Y(J,2)*Y(I,1)+Y(J,1)*Y(I,2))
           +       +CD(2)*CD(3)*(Y(J,3)*Y(I,2)+Y(J,2)*Y(I,3))
           +       +CD(3)*CD(1)*(Y(J,1)*Y(I,3)+Y(J,3)*Y(I,1))

               CSN=CD(3)*(Y(J,2)*Y(I,1)-Y(J,1)*Y(I,2))
           +      +CD(2)*(Y(J,1)*Y(I,3)-Y(J,3)*Y(I,1))
           +      +CD(1)*(Y(J,3)*Y(I,2)-Y(J,2)*Y(I,3))

               DERA1=-2*SIN(DELTA)*(CCS1+CCS2)+2*COS(DELTA)*CSN
               DERA2=-2*COS(DELTA)*(CCS1+CCS2)-2*SIN(DELTA)*CSN
            ELSE
               DERA1=0.D0
               DERA2=0.D0
            ENDIF
C
C...        Si les atomes Z(I,.) et Z(J,.) sont au moins 1,4 et si ils sont
C...        suffisamment proches, on calcule la derivee de leurs energies.
C
            IF ((DIST(I,J).GE.3).AND.((SQRT(RACAR)).LE.9.D0)) THEN

               ECOUL2=ECOUL2-QC(I)*QC(J)*0.5*332.17*(DERA2*RACAR
           +                -3/2*DERA1*DERA1)/(SQRT(RACAR)*RACAR*RACAR)

               CTS=-1/0.0736/(RVDW(I)+RVDW(J))

               EVDW2=EVDW2+EPS(I,J)*( 3*2.25*((RVDW(I)+RVDW(J))**6)
           +          *( DERA2*RACAR-4*DERA1*DERA1 )/(RACAR**5)
           +          +8.28D5*0.5*CTS*( DERA2*EXP(CTS*SQRT(RACAR))
           +          *SQRT(RACAR)+EXP(CTS*SQRT(RACAR))*0.5*CTS
           +          *DERA1*DERA1-EXP(CTS*SQRT(RACAR))
           +          *0.5*DERA1*DERA1/SQRT(RACAR) )/RACAR )

            ENDIF

40          CONTINUE
50       CONTINUE


         RETURN
         END
C
C -------------------------------------------------------------------------
C -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE LECDONBIS
      ====================
c
c
c     Description :
c     -------------
c
c     Cette procedure lit les atomes de la molecule de reference et de la
c     molecule mobile a egaler (AFE et AME) et les atomes de la molecule
c     mobile definissant l'axe autour duquel on effectue le fitting (AXE).
c
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /LIAISO/  AXE
      COMMON  /DIVISI/  NBATEG,AME,AFE

      INTEGER       AXE(4),
     1              NBATEG,AME(NDIM1),AFE(NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER       I,J
      LOGICAL       TEST,LIAD
c
c
c     Corps :
c     -------
c
c
c...  TEST est une variable verifiant les donnees introduites dans cette
c...  routine.
c
      TEST=.FALSE.

      DO 20 WHILE (.NOT.TEST)

         WRITE(*,1100)
c
c...     Lecture du nombre d'atomes a egaler.
c
         WRITE(*,1200)
         READ (*,*) NBATEG
c
c...     Lecture des atomes de la molecule de reference a faire coincider
c...     ceux de la molecule mobile.
c
         WRITE(*,1300) NBATEG
         READ (*,*) (AFE(I),I=1,NBATEG)
c
c...     Lecture des atomes de la molecule de mobile a faire coincider
c...     ceux de la molecule reference.
c
         WRITE(*,1400) NBATEG
         READ (*,*) (AME(I),I=1,NBATEG)
```

```
c
c...        Lecture des atomes de la molecule mobile definissant l'axe de
c...        rotation.
c
            WRITE(*,1600)
            READ (*,*) (AXE(J),J=1,4)
c
c...        Si la fonction logique LIAD est fausse, cela signifie que la
c...        liaison autour de laquelle on effectue le fitting flexible
c...        (definie par AXE), n'est pas admissible.
c
            TEST=LIAD(AXE)

20        CONTINUE


          RETURN
c
c         Formats :
c         ---------
c
1000      FORMAT(A)
1100      FORMAT(1X,T5'Introduisez les atomes a comparer et l''axe.',/)
1200      FORMAT(1X,T5,'Entrez le nombre d''atomes a comparer : ',$)
1300      FORMAT(1X,T5,'Entrez les ',I2,' atomes a egaler de la molecule'
          1       ,' fixe :   ',$)
1400      FORMAT(1X,T5,'Entrez les ',I2,' atomes a egaler de la molecule'
          1       ,' mobile : ',$)
1600      FORMAT(1X,T5,'Entrez l''axe de rotation : ',$)

          END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE LECDON
c     =================
c
c
c     Description :
c     -------------
c
c     Cette procedure lit les atomes de la molecule de reference et de la
c     molecule mobile a egaler (AFE et AME) et les atomes de la molecule
c     mobile definissant les axes autour desquels on effectue le fitting
c     (AXE).
c
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NAXE  = 10

      COMMON  /LIAISO/  NBRAXE,AXE
      COMMON  /DIVISI/  NBATEG,AME,AFE

      INTEGER        NBRAXE,AXE(4,NAXE),
     1               NBATEG,AME(0:NDIM1,0:NAXE),AFE(NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER        I,J
      LOGICAL        TEST,MULLIA
c
c
c     Corps :
c     -------
c
c
c...  TEST est une variable verifiant les donnees introduites dans cette
c...  routine.
c
      TEST=.FALSE.

      DO 20 WHILE (.NOT.TEST)

         WRITE(*,1100)
c
c...     Lecture du nombre d'atomes a egaler.
c
         WRITE(*,1200)
         READ (*,*) NBATEG
         AME(0,0)=NBATEG
```

```
c
c...        Lecture des atomes de la molecule de reference a faire coincider
c...        ceux de la molecule mobile.
c
            WRITE(*,1300) NBATEG
            READ (*,*) (AFE(I),I=1,NBATEG)
c
c...        Lecture des atomes de la molecule mobile a faire coincider avec
c...        ceux de la molecule reference.
c
            WRITE(*,1400) NBATEG
            READ (*,*) (AME(I,0),I=1,NBATEG)
c
c...        Lecture du nombre d'axes de rotation et des atomes de la molecule
c...        mobile definissant ceux-ci.
c
            WRITE(*,1500)
            READ(*,*) NBRAXE
            DO 10 I=1,NBRAXE
               WRITE(*,1600) I
               READ (*,*) (AXE(J,I),J=1,4)
10          CONTINUE
c
c...        Test verifie dans un premier temps que le nombre d'axes de rotation
c...        est plus petit ou egal au nombre d'atomes a egaler.
c...        Cette hypothese est due a la routine d'optimisation utilisee.
c
            TEST=NBRAXE.LE.NBATEG

            IF (.NOT.TEST) THEN
               CALL SCLEAR
               WRITE(*,1700)
            ELSE
c
c...           Si la fonction logique MULLIA est fausse, cela signifie qu'une
c...           liaison autour de laquelle on effectue le fitting flexible
c...           (definie par AXE), n'est pas admissible.
c
               TEST=MULLIA()

            ENDIF

20       CONTINUE


         RETURN
```

```
c
c       Formats :
c       ---------
c
1000    FORMAT(A)
1100    FORMAT(1X,T5'Introduisez les atomes a comparer et les axes.',/)
1200    FORMAT(1X,T5,'Entrez le nombre d''atomes a comparer : ',$)
1300    FORMAT(1X,T5,'Entrez les ',I2,' atomes a egaler de la molecule'
        1       ,' fixe :   ',$)
1400    FORMAT(1X,T5,'Entrez les ',I2,' atomes a egaler de la molecule'
        1       ,' mobile : ',$)
1500    FORMAT(1X,T5,'Entrez le nombre d''axes : ',$)
1600    FORMAT(1X,T5,'Entrez le ',I2,' eme axe : ',$)
1700    FORMAT(1X,79('.'),/,1X,'.',T30,'A T T E N T I O N',T80,'.',
        1       /,1X,'.',T3,'Le nombre d''axes de rotation est plus ',
        2       'grand que le nombre d''atomes a comparer',T80,'.',/,
        3       1X,79('.'),/)

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE DONNEE (MOL,MAT,NAT,NLIEN)
c     =================
c
c     Description :
c     -------------
c
c     Cette procedure lit un fichier de donnees contenant le nombre d'atomes
c     NAT, le nombre de liaisons NLIEN, l'ensemble des paires d'atomes lies
c     BOND, l'ensemble des coordonnees de chaque atome MAT, le nombre
c     atomique NBAT, le type ITYP et la charge CHGE de chaque atome.
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100

      INTEGER           NAT,NLIEN
      DOUBLE PRECISION  MAT(NDIM1,3)
      CHARACTER*(*)     MOL
c
c     Commons :
c     .........
c
      PARAMETER  NDIM2 = 200

      COMMON  /MOLECU/  BOND,NBAT,ITYP,CHGE,TITRE

      INTEGER           BOND(2,NDIM2),NBAT(NDIM1),ITYP(NDIM1)
      DOUBLE PRECISION  CHGE(NDIM1)
      CHARACTER*70      TITRE
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
      CHARACTER*70      NOM
      LOGICAL           TEST,VERLI,VERDEG,VERMS,VERSC
```

```
c
c
c       Corps :
c       -------
c
c
c...    Test est une variable verifiant les donnees introduites dans cette
c...    routine.
c
        TEST=.FALSE.

        DO 10 WHILE(.NOT.TEST)
c
c...       Lecture du nom du fichier contenant les donnees.
c
           WRITE(*,1100) MOL
           READ(*,1000) NOM

           OPEN (UNIT=20,FILE=NOM,STATUS='OLD')
c
c...       Lecture du titre du fichier.
c
           READ(20,1000) TITRE
c
c...       Lecture du nombre de liaisons et lecture des paires d'atomes lies.
c
           READ(20,*) NLIEN
           READ(20,*) (BOND(1,j),BOND(2,j),J=1,NLIEN)
c
c...       Lecture du nombre d'atomes.
c
           READ(20,*) NAT
c
c...       Pour chaque atome, lecture du nombre atomique, des coordonnees,
c...       du type et de la charge.
c
           READ(20,*) (NBAT(I),(MAT(I,J),J=1,3),ITYP(I),CHGE(I),I=1,NAT)

           CLOSE(UNIT=20)
c
c...       Si VERDEG est faux, cela signifie qu'il existe au moins un atome
c...       de la molecule qui est lie a plus de quatre autres atomes de
c...       cette molecule. C'est contraire aux hypotheses du programme.
c
           TEST=VERDEG(NAT,NLIEN,BOND)
           IF (TEST) THEN
c
c...          Si VERDEG est vrai, alors verifions que la molecule forme un
c...          multigraphe simple i.e. qu'aucun atome n'est lie a lui-meme et
c...          qu'il n'existe pas de liaison identique.
c
              TEST=VERMS(NAT)
```

```
c
c...          Si VERMS est vrai, alors verifions que la molecule forme un
c...          multigraphe simplement connexe i.e. que tous les atomes sont lies
c
              IF (TEST) TEST=VERSC(NAT,NLIEN,BOND)
            ENDIF

10      CONTINUE


        RETURN
c
c       Formats :
c       ---------
c
1000    FORMAT(A)
1100    FORMAT(/,1X,T5,'Introduisez le nom du fichier contenant la',
        1       ' molecule ',A,'.',/,1X,T5,'NOM : ',$)

        END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE INIVDW (NAT,NLIEN,NBAT,BOND,RVDW,EPS)
c     ================
c
c     Description :
c     -------------
c
c     Cette procedure initialise les parametres de l'energie de Van Der Waals
c     selon la molecule donnee par NAT(le nombre d'atomes), NLIEN(le nombre
c     de liaisons), BOND(l'ensemble des paires d'atomes lies) et NBAT(le nom-
c     bre atomique de chaque atome) : RVDW(I) sera le rayon de Van Der Waals
c     du le atome et EPS(I,J) sera la constante de Van Der Waals associee aux
c     atomes I et J.
c     (cfr. structure du fichier VDWFF.DAT)
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      INTEGER           NAT,NLIEN,NBAT(NDIM1),BOND(2,NDIM2)
      DOUBLE PRECISION  RVDW(NDIM1),EPS(NDIM1,NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J,NM(NDIM1)
      DOUBLE PRECISION  RS(10),E(10,10)
c
c
c     Corps :
c     -------
c
      OPEN(UNIT=20,NAME='VDWFF',STATUS='OLD')
c
c...  Lecture de l'ensemble des rayons de Van Der Waals.
c
      READ(20,1000) (RS(J),J=1,10)
c
c...  Lecture de l'ensemble des constantes de Van Der Waals.
c
      DO 10 I=1,10
         READ(20,1100) (E(I,J),J=1,10)
10    CONTINUE
c
      DO 30 I=1,10
         DO 20 J=1,10
            E(J,I)=E(I,J)
20       CONTINUE
30    CONTINUE
c
      CLOSE (UNIT=20)
```

```
c
c...    Recherche, parmi l'ensemble des rayons de Van Der Waals et l'ensemble
c...    des constantes de Van Der Waals, des parametres necessaires.
c

        DO 50 I=1,NAT
           IF ((NBAT(I).EQ.15).OR.(NBAT(I).EQ.16)) THEN
              NM(I)=6
           ELSEIF (NBAT(I).EQ.17) THEN
              NM(I)=7
           ELSEIF (NBAT(I).EQ.35) THEN
              NM(I)=8
           ELSEIF (NBAT(I).EQ.53) THEN
              NM(I)=9
           ELSEIF ((NBAT(I).GE.5).AND.(NBAT(I).LE.9)) THEN
              NM(I)=NBAT(I)-4
           ELSEIF ((NBAT(I).GE.1).AND.(NBAT(I).LE.4)) THEN
              NM(I)=1
              DO 40 J=1,NLIEN
                 IF (BOND(1,J).EQ.I) THEN
                    IF ((NBAT(BOND(2,J)).EQ.7).OR.
     +                          (NBAT(BOND(2,J)).EQ.8)) THEN
                       NM(I)=10
                    ELSEIF (BOND(2,J).EQ.I) THEN
                       IF ((NBAT(BOND(1,J)).EQ.7).OR.
     +                     (NBAT(BOND(1,J)).EQ.8)) THEN
                          NM(I)=10
                       ENDIF
                    ENDIF
                 ENDIF
40            CONTINUE
           ENDIF
50      CONTINUE

        DO 70 I=1,NAT
           RVDW(I)=RS(NM(I))
           DO 60 J=I+1,NAT
              EPS(I,J)=E(NM(I),NM(J))
              EPS(J,I)=EPS(I,J)
60         CONTINUE
70      CONTINUE


1000    FORMAT(10F4.2)
1100    FORMAT(10F5.3)


        RETURN
        END
c
c -------------------------------------------------------------------------
c -------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE INITOR (AXE,ITYP,VT,NT,ST)
      =================
c
c
c     Description :
c     -------------
c
c     Cette procedure initialise les parametres de l'energie de torsion selon
c     l'AXE de rotation du fitting, et le type des atomes definissant cet axe
c     Ces parametres sont VT, la barriere de torsion, NT, ST, le signe de NT.
c     (cfr. structure du fichier TORFF.DAT)
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100

      INTEGER           AXE(4),ST(9),ITYP(NDIM1)
      DOUBLE PRECISION  VT(9),NT(9)
c
c     Commons :
c     .........
c
      COMMON  /ADJACENCE/  ADJL

      INTEGER           ADJL(0:4,NDIM1)
c
c     Variables internes :
c     ....................
c
      PARAMETER  DIM = 100

      INTEGER           CAS,TYP1(DIM),TYP2(DIM),I,L,M,N
      DOUBLE PRECISION  VO(DIM),FD(DIM)
      LOGICAL           TEST
c
c
c     Corps :
c     -------
c
      OPEN(UNIT=20,NAME='TORFF.DAT',STATUS='OLD')
c
c...  Lecture du fichier TORFF.DAT contenant les parametres recherches.
c
      READ(20,*) CAS

      DO 10 I=1,CAS
         READ(20,*) TYP1(I),TYP2(I),VO(I),FD(I)
10    CONTINUE

      CLOSE(UNIT=20)
```

```
c
c...    Recherche des parametres VT, NT, ST : les parametres VT et NT sont
c...    donnes dans le fichier TORFF.DAT. Ils dependent du type des deux atomes
c...    definissant l'axe de rotation.
c
        L=1
        DO 40 M=1,ADJL(0,AXE(2))
           IF (ADJL(M,AXE(2)).NE.AXE(3)) THEN
              DO 30 N=1,ADJL(0,AXE(3))
                 IF (ADJL(N,AXE(3)).NE.AXE(2)) THEN
                    I=0
                    TEST=.FALSE.
                    DO 20 WHILE ((I.LT.CAS).AND.(.NOT.TEST))
                       I=I+1
                       TEST=(((ITYP(AXE(2)).EQ.TYP1(I)).AND.(ITYP(AXE(3))
        1                  .EQ.TYP2(I))).OR.((ITYP(AXE(3)).EQ.TYP1(I)).AND.
        2                  (ITYP(AXE(2)).EQ.TYP2(I))))
20                  CONTINUE

                    VT(L)=VO(I)
                    NT(L)=FD(I)
                    ST(L)=NT(L)/ABS(NT(L))

                    L=L+1

                 ENDIF
30            CONTINUE
           ENDIF
40      CONTINUE


        RETURN
        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
c
      SUBROUTINE INICOUL (YAT,NBAT,CHGE,QC)
      ==================
c
c
c     Description :
c     -------------
c
c     Cette procedure initialise  les parametres de l'energie de Coulomb a
c     savoir QC(i), la charge nette de l'atome i de la molecule Y possedant
c     YAT atomes dont le nombre atomique est NBAT(i) et dont la charge totale
c     est CHGE(i).
c
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      PARAMETER  NDIM1 = 100

      INTEGER           YAT,NBAT(NDIM1)
      DOUBLE PRECISION  QC(NDIM1),CHGE(NDIM1)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,NCH
c
c
c     Corps :
c     -------
c
      I=1

      DO 10 WHILE((I.LE.YAT).AND.(CHGE(I).NE.0))
        NCH=NBAT(I)-2
        IF (NCH.LE.0) NCH=1
        IF (NBAT(I).GT.10) NCH=NCH-8
        QC(I)=NCH-CHGE(I)
        I=I+1
10    CONTINUE


      RETURN
      END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE ECRITU
      =================
c
c
c     Description :
c     -------------
c
c     Cette procedure ecrit les resultats du fitting sur un fichier.
c
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NDIM2 = 200

      COMMON  /MOBILE/  YAT,Y,YLIEN
      COMMON  /MOLECU/  BOND,NBAT,ITYP,CHGE,TITRE

      INTEGER           YLIEN,BOND(2,NDIM2),YAT,ITYP(NDIM1),NBAT(NDIM1)
      DOUBLE PRECISION  Y(NDIM1,3),CHGE(NDIM1)
      CHARACTER*70      TITRE
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
      CHARACTER*70      NOM
c
c
c     Corps :
c     -------
c
c...  Lire le nom du fichier.
c
      WRITE(*,1100)
      READ(*,1000) NOM

      OPEN (UNIT=20,FILE=NOM,STATUS='NEW')
c
c...  Ecrire le titre du fichier.
c
      WRITE(20,1200) TITRE
c
c...  Ecriture du nombre de liaisons de la molecule mobile Y.
c
      WRITE(20,1300) YLIEN
c
c...  Ecrire les paires d'atomes definissant les liaisons.
c
      WRITE(20,1400) (BOND(1,I),BOND(2,I),I=1,YLIEN)
```

```
c
c...    Ecriture du nombre d'atomes de la molecule mobile Y.
c
        WRITE(20,1300) YAT
c
c...    Pour chaque atome, ecrire le nombre atomique, les coordonnees,
c...    le type et la charge.
c
        WRITE(20,1500) (NBAT(I),(Y(I,J),J=1,3),ITYP(I),CHGE(I),I=1,YAT)

        CLOSE (UNIT=20)


        RETURN
c
c       Formats :
c       ---------
c
1000    FORMAT(A)
1100    FORMAT(1X,T5,'Introduisez le nom du fichier de sortie de la',
        1       ' molecule mobile.',/,T5,'NOM : ',$)
1200    FORMAT(1X,A70)
1300    FORMAT(1X,I10)
1400    FORMAT(1X,2013)
1500    FORMAT(1X,I6,3F10.5,4X,i4,F7.3)

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE IMPATO
c     =================
c
c
c     Description :
c     -------------
c
c     Cette routine imprime les coordonnees des atomes des molecules fixe
c     et mobile sur le fichier des iterees.
c
c     Declarations :
c     --------------
c
c     Commons :
c     .........
c
      PARAMETER NDIM1 = 100

      COMMON  /FIXE/    XAT,X
      COMMON  /MOBILE/  YAT,Y

      INTEGER           XAT,YAT
      DOUBLE PRECISION  X(NDIM1,3),Y(NDIM1,3)
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
c
c
c     Corps :
c     -------
c
c...  Impression d'un titre sur le fichier.
c
      WRITE(90,2150)
c
c...  Impression des YAT premiers atomes des deux molecules puis des atomes
c...  restant de la molecule fixe.
c
      IF (XAT.GT.YAT) THEN
         WRITE(90,2200) (I,(X(I,J),J=1,3),(Y(I,J),J=1,3),I=1,YAT)
         WRITE(90,2230) (I,(X(I,J),J=1,3),I=YAT+1,XAT)
c
c...  Impression des XAT premiers atomes des deux molecules puis des atomes
c...  restant de la molecule mobile.
c
      ELSE
         WRITE(90,2200) (I,(X(I,J),J=1,3),(Y(I,J),J=1,3),I=1,XAT)
         WRITE(90,2260) (I,(Y(I,J),J=1,3),I=XAT+1,YAT)
      ENDIF


      RETURN
```

```
c
c       Formats :
c       ---------
c
2150    FORMAT(1X,T5,'NO',T21,'Molecule fixe',T57,'Molecule mobile',/,
        1      1X,T4,4('-'),T20,15('-'),T56,17('-'))
2200    FORMAT(1X,T5,I2,T12,3F10.5,T49,3F10.5)
2230    FORMAT(1X,T5,I2,T12,3F10.5)
2260    FORMAT(1X,T5,I2,T49,3F10.5)

        END
c
c -------------------------------------------------------------------------------
c -------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------
c ------------------------------------------------------------------------
c
      SUBROUTINE IMPEGA (TEST,FNORM)
c     ==============================
c
c     Description :
c     -------------
c
c     Cette routine imprime les atomes a comparer et la somme des distances
c     au carre sur le fichier des iterees.
c     De plus elle calcule le rms FNORM.
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      LOGICAL           TEST
      DOUBLE PRECISION  FNORM
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100
      PARAMETER  NAXE  = 10

      COMMON  /FIXE/    XAT,X
      COMMON  /MOBILE/  YAT,Y
      COMMON  /DIVISI/  NBATEG,AME,AFE

      INTEGER           XAT,YAT,
      1                 NBATEG,AFE(NDIM1),AME(0:NDIM1,0:NAXE)
      DOUBLE PRECISION  X(NDIM1,3),Y(NDIM1,3),
      1                 NORM(NDIM1),FCT
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
c
c
c     Corps :
c     -------
c
c...  Cette procedure calcule la distance entre chaque atome (NORM),
c...  la somme des distances au carre (FCT) et le rms (FNORM).
c
      CALL RMS(NORM,FCT,FNORM)
c
c...  Impression d'un titre.  Il est souligne si TEST est vrai.
c
      WRITE(90,2450)
      IF (TEST) THEN
         WRITE(90,2850)
      ENDIF
      WRITE(90,*)
```

```
c
c...    Impression de chaque paire d'atome a comparer et de la distance les
c...    separant.
c
        WRITE(90,2500) (AFE(I),(X(AFE(I),J),J=1,3),AME(I,0),
        1               (Y(AME(I,0),J),J=1,3),NORM(I),I=1,NBATEG)
c
c...    Impression de la somme des distances au carre (FCT).
c
        WRITE(90,2550) FCT


        RETURN
c
c       Formats :
c       ---------
c
2450    FORMAT(1X,T5,'NO',T19,'Molecule fixe',T44,'NO',
        1      T57,'Molecule mobile',T84,'Distance')
2500    FORMAT(1X,T5,I2,3X,3F10.5,T44,I2,3X,3F10.5,T82,F10.6)
2550    FORMAT(/,1X,T5,'La somme des distances au carre : ',F10.6)
2850    FORMAT(1X,T4,4('-'),T18,15('-'),T43,4('-'),T56,17('-'),
        1      T83,10('-'))

        END
c
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
```

```
c ------------------------------------------------------------------------------
c ------------------------------------------------------------------------------
c
      SUBROUTINE IMPEGABIS (TEST,FNORM)
c     =================================
c
c     Description :
c     -------------
c
c     Cette routine imprime les atomes a comparer et la somme des distances
c     au carre sur le fichier des iterees.
c     De plus elle calcule le rms FNORM.
c
c     Declarations :
c     --------------
c
c     Parametres :
c     ............
c
      LOGICAL           TEST
      DOUBLE PRECISION  FNORM
c
c     Commons :
c     .........
c
      PARAMETER  NDIM1 = 100

      COMMON  /FIXE/    XAT,X
      COMMON  /MOBILE/  YAT,Y
      COMMON  /DIVISI/  NBATEG,AME,AFE

      INTEGER           XAT,YAT,
     1                  NBATEG,AFE(NDIM1),AME(NDIM1)
      DOUBLE PRECISION  X(NDIM1,3),Y(NDIM1,3),
     1                  NORM(NDIM1),FCT
c
c     Variables internes :
c     ....................
c
      INTEGER           I,J
c
c
c     Corps :
c     -------
c
c...  Cette procedure calcule la distance entre chaque atome (NORM),
c...  la somme des distances au carre (FCT) et le rms (FNORM).
c
      CALL RMSBIS(NORM,FCT,FNORM)
c
c...  Impression d'un titre.  Il est souligne si TEST est vrai.
c
      WRITE(90,2450)
      IF (TEST) THEN
         WRITE(90,2850)
      ENDIF
      WRITE(90,*)
```

```
c
c...    Impression de chaque paire d'atome a comparer et de la distance les
c...    separant.
c
        WRITE(90,2500) (AFE(I),(X(AFE(I),J),J=1,3),AME(I),
        1               (Y(AME(I),J),J=1,3),NORM(I),I=1,NBATEG)
c
c...    Impression de la somme des distances au carre (FCT).
c
        WRITE(90,2550) FCT


        RETURN
c
c       Formats :
c       ---------
c
2450    FORMAT(1X,T5,'NO',T19,'Molecule fixe',T44,'NO',
        1      T57,'Molecule mobile',T84,'Distance')
2500    FORMAT(1X,T5,I2,3X,3F10.5,T44,I2,3X,3F10.5,T82,F10.6)
2550    FORMAT(/,1X,T5,'La somme des distances au carre : ',F10.6)
2850    FORMAT(1X,T4,4('-'),T18,15('-'),T43,4('-'),T56,17('-'),
        1      T83,10('-'))

        END
c
c ---------------------------------------------------------------------------
c ---------------------------------------------------------------------------
```

```
C ------------------------------------------------------------------------------
C ------------------------------------------------------------------------------
C
      SUBROUTINE PRESEN
      =================
C
C
C     Description :
C     -------------
C
C     Cette routine affiche le titre du programme a l'ecran.
C
C     Corps :
C     -------
C
      CALL SCLEAR
      WRITE(*,1000)
      WRITE(*,1100)
      WRITE(*,1200)
      CALL  WAIT (15.0)


      RETURN
C
C     Formats :
C     ---------
C
1000  FORMAT(1X,T2,77('*'),/,T2,'*',T78,'*')
1100  FORMAT(1X,T2,'*',T34,'E F M F I T',T78,'*',/,1X,T2,'*',T33
     1      ,18('-'),T78,'*',2(/,1X,T2,'*',T78,'*'))
1200  FORMAT(1X,T2,'*',T21,'Techniques de comparaison moleculaire',T78,
     1      '*',/,1X,T2,'*',T7,'tenant compte de l''energie de ',
     2      'contrainte de la molecule mobile ou',T78,'*',/,1X,T2,'*',
     3      T7,'permettant de considerer plusieurs liaisons comme ',
     4      'axe de rotation.',T78,'*',/,9(1X,T2,'*',T78,'*',/),
     5      1X,T2,'*',T5,'Developpe par',T30,'Dans le ',
     6      'cadre du',T78,'*',/,1X,T2,'*',T78,'*',/,1X,T2,'*',T8,
     7      'P Culot',T59,'memoire de licence',T78,'*',/,1X,T2,'*',T8,
     8      'X Gillo',T59,' en mathematique',T78,'*',/,1X,T2,'*',
     9      T78,'*',/,1X,T2,77('*'))

      END
C
C ------------------------------------------------------------------------------
C ------------------------------------------------------------------------------
```

Annexe

Angle de torsion

Les méthodes de comparaison moléculaire que nous avons développées nécessitent le calcul de l'angle de torsion. Les chimistes disposent dans leur programme IFMFIT d'une procédure TORANG qui calcule cet angle de torsion [13]. Etant en possession d'un formalisme propre présenté dans cette annexe, nous l'avons adapté au problème. Les deux résultats étant identiques, nous avons conservé notre formalisme.

Avant d'expliquer notre méthode, rappelons que l'angle de torsion est déterminé par quatre atomes I, J, K et L. L'angle de torsion est l'angle formé par les plans α et β où le plan α est défini par le triplet (I, J, K) et le plan β est défini par le triplet (J,K,L).

A.I Rappels théoriques

Un plan dans l'espace peut être défini par trois points non alignés, ou par deux points et un vecteur qui n'est pas parallèle au segment qui joint les deux points, ou encore par deux vecteurs non parallèles et un point.

Supposons que le plan soit donné par trois points non alignés : $P1(x_1,y_1,z_1)$, $P2(x_2,y_2,z_2)$ et $P3(x_3,y_3,z_3)$. Dès lors, une <u>représentation paramétrique</u> du plan est

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + \begin{pmatrix} x_2-x_1 \\ y_2-y_1 \\ z_2-z_1 \end{pmatrix} . \tau + \begin{pmatrix} x_3-x_1 \\ y_3-y_1 \\ z_3-z_1 \end{pmatrix} . t$$

L'<u>équation cartésienne</u> du plan s'écrit $Ax+By+Cz+D=0$.

Cette équation s'obtient de la représentation paramétrique en résolvant

$$\begin{vmatrix} x-x_1 & v_1 & w_1 \\ y-y_1 & v_2 & w_2 \\ z-z_1 & v_3 & w_3 \end{vmatrix} = 0$$

c'est-à-dire,

$$(x-x_1).\underbrace{(v_2.w_3-v_3.w_2)}_{A}+(y-y_1).\underbrace{(v_3.w_1-v_1.w_3)}_{B}+(z-z_1).\underbrace{(v_1.w_2-v_2.w_1)}_{C}=0$$

ou encore,

$$Ax+By+Cz+\underbrace{(-Ax_1-By_1-Cz_1)}_{D}=0$$

Nous obtenons finalement la forme normale de l'équation d'un plan :

$$n_1x+n_2y+n_3z+p=0 \text{ avec } \|\vec{N}\| = 1 \quad \text{où} \quad \vec{N}=\begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix}$$

Pour cela, il suffit de diviser l'équation cartésienne par $\sqrt{A^2+B^2+C^2}$:

$$n_1=A/\sqrt{A^2+B^2+C^2} \qquad n_3=C/\sqrt{A^2+B^2+C^2}$$
$$n_2=B/\sqrt{A^2+B^2+C^2} \qquad p=D/\sqrt{A^2+B^2+C^2}$$

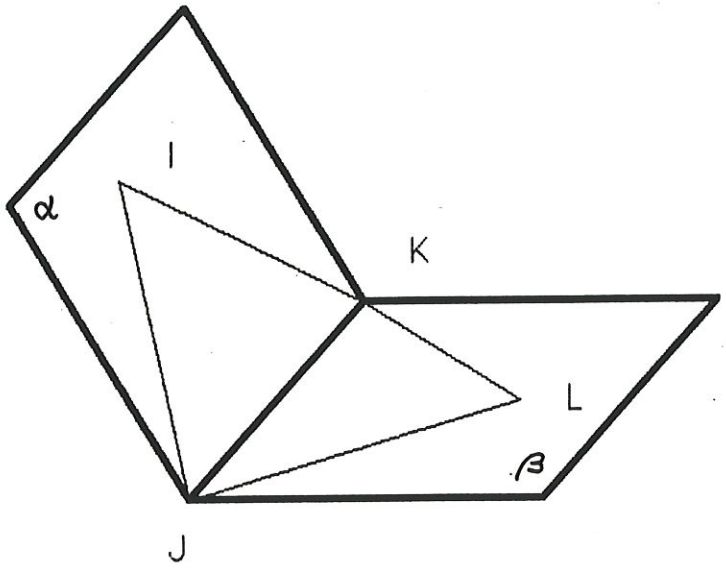
Si on considère deux plans d'équation normale respective $\vec{n}.\vec{x}=-\vec{p}$ et $\vec{n}^*.\vec{x}^*=-\vec{p}^*$, l'angle φ qu'ils forment est égal à l'angle formé par leurs vecteurs normaux : $\cos(\varphi)=\vec{n}.\vec{n}^*$.

A2 Calcul de l'angle de torsion

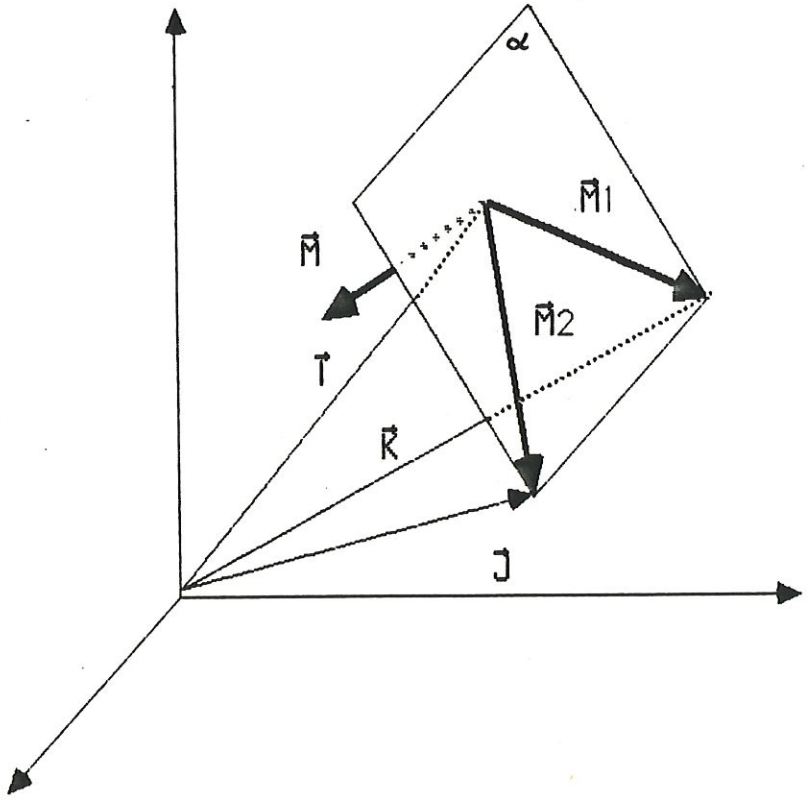
Ces rappels nous indiquent que pour calculer l'angle de torsion, il suffit de déterminer le vecteur normal du plan α et celui du plan β et de calculer l'arc cosinus du produit scalaire de ces deux vecteurs normaux.

Toutefois, cette méthode ne nous renseigne que sur la valeur absolue de l'angle de torsion c'est-à-dire $0 \leq \varphi \leq \pi$. Nous devons encore rechercher le signe de cet angle c'est-à-dire $-\pi \leq \varphi \leq \pi$.

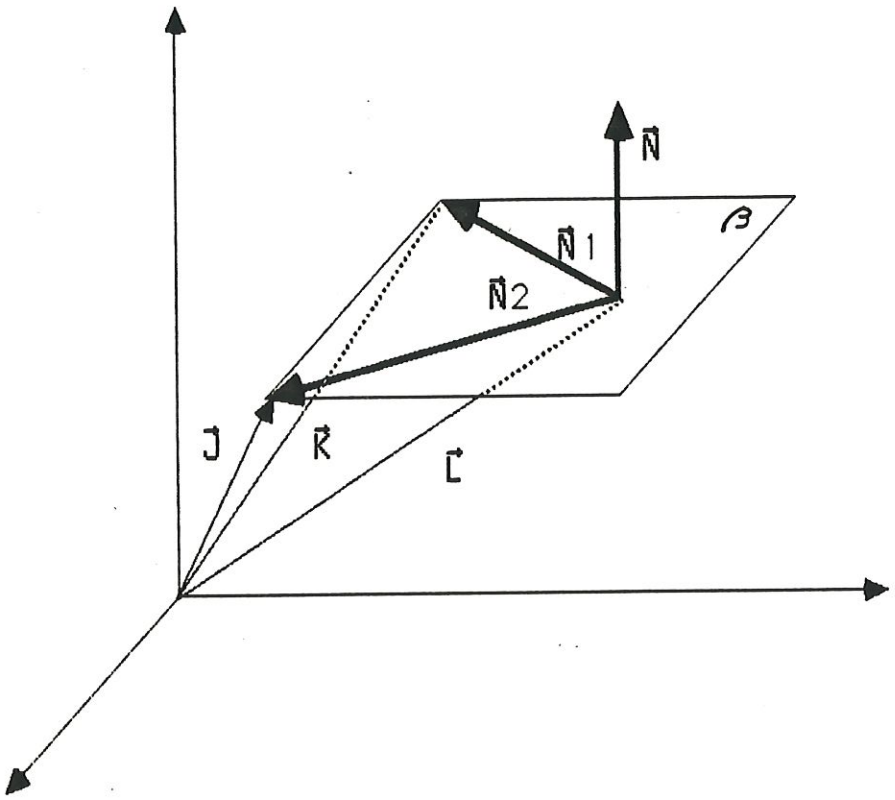
Sans perte de généralité, considérons les plans α et β comme suit :



Soit $\vec{M}$ le vecteur normal du plan α déterminé par le produit vectoriel des vecteurs $\vec{M}_1=\vec{R}-\vec{T}$ et $\vec{M}_2=\vec{J}-\vec{T}$.



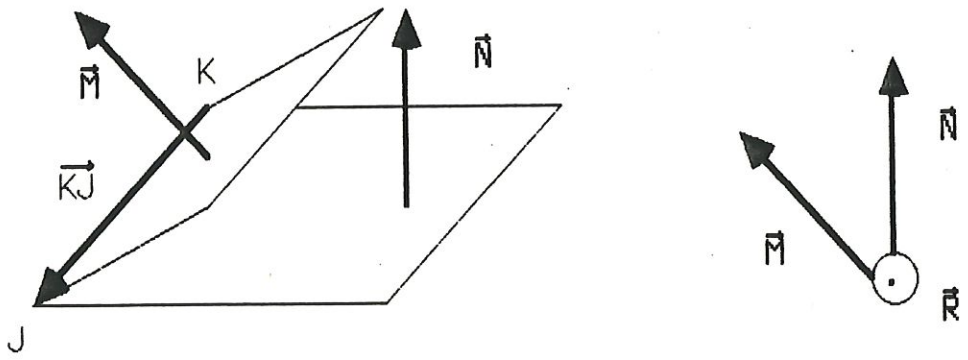
Soit $\vec{N}$ le vecteur normal du plan β déterminé par le produit vectoriel des vecteurs $\vec{N}_1=\vec{K}-\vec{L}$ et $\vec{N}_2=\vec{J}-\vec{I}$.



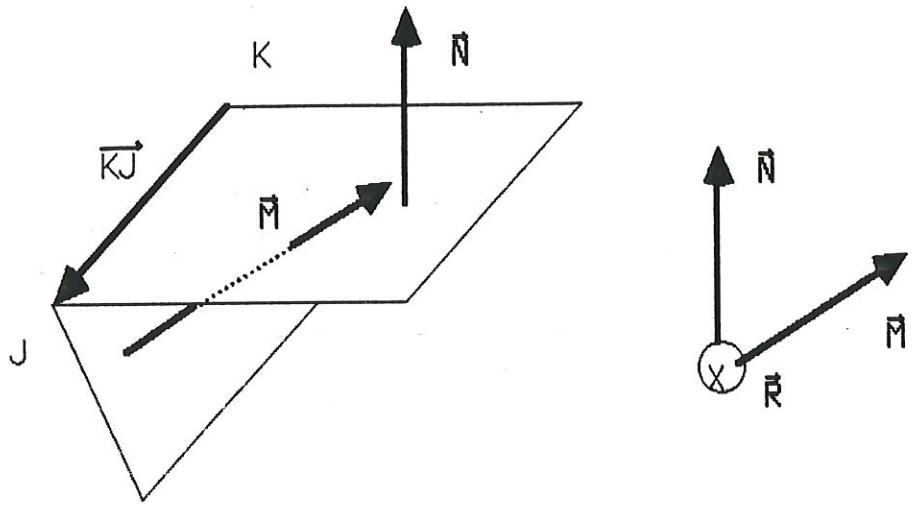
Remarque

Si les deux plans sont confondus, c'est-à-dire si I=J, nous avons bien $\vec{M}=\vec{N}$.

Soit $\vec{R}=\vec{N}x\vec{M}$



Nous voyons que si l'angle entre les deux plans est compris entre 0 et π, le vecteur $\vec{R}$ a le même sens que le vecteur $\overrightarrow{KJ}$.



Nous voyons que si les deux plans forment un angle compris entre 0 et $-\pi$, le vecteur $\vec{R}$ est de sens opposé à celui du vecteur $\overrightarrow{KJ}$.

Ainsi, pour déterminer le signe de l'angle formé par les plans α et β, nous devons calculer le produit vectoriel des vecteurs normaux de ces plans et comparer le sens de ce vecteur à celui du vecteur $\vec{J}$-$\vec{R}$: si ces vecteurs sont de même sens, l'angle φ est positif; sinon φ est négatif.

Conclusions

L'importance des techniques de comparaison moléculaire en pharmacologie n'est plus à démontrer. Deux programmes, appelés fitting rigide et flexible, ont été développés dans ce but par le Laboratoire de Chimie Moléculaire Structurale et Radiocristallographie. Aucun ne tient cependant compte de notions d'énergie interne conformationnelle , un concept cependant important.

Nous avons pour cela développé trois méthodes de comparaison moléculaire tenant compte de notions d'énergie basées sur la mécanique moléculaire.

La première consiste à résoudre un problème de minimisation d'une fonction non linéaire, la somme des distances au carré entre les atomes à comparer, sous une contrainte non linéaire, l'énergie de conformation de la molécule.

La deuxième résoud le problème de minimisation d'une fonction non linéaire, la somme des distances au carré entre les atomes à comparer, sous une contrainte de bornes. Ces bornes définissent un intervalle sur lequel l'énergie de conformation est admissible.

La troisième recherche l'angle minimisant la somme des distances au carré pour lequel le passage de barrières énergétiques, s'il y en a, pour aboutir à la conformation optimale est tel que chaque apport ne dépasse pas une quantité seuil définie par l'utilisateur ("le bon sens chimique").

De plus, nous développons une méthode considérant plusieurs liaisons interatomiques simples comme axes de rotation.

Cette méthode consiste à résoudre un problème de moindres carrés, soit minimiser la somme des distances au carré entre les atomes à comparer.

En outre, nous avons élaboré plusieurs algorithmes vérifiant la connexité ainsi que l'admissibilité des liaisons prises comme axes de rotation.

Ces algorithmes utilisent, pour effectuer ces tests, la matrice d'adjacence, la recherche en profondeur et le palmier créé à partir de la molécule prise comme un multigraphe.

Les méthodes citées ci-dessus pour ces problèmes de minimisation donnent des résultats fiables et l'introduction des tests de connexité et d'admissibilité de la liaison permet une interactivité entre le programme de superposition moléculaire et l'utilisateur.

Références

Références

1. L. Barino et R. Scordamaglia, "Characterization and predictive evaluation of active compound through molecular fit techniques", La chimica e l'industria, **68**, 118 (1986)

2. S.C. Nyburg, "Some uses of a best molecular fit routine", Acta Crystallogr., **B30**, 251 (1974)

3. P.S Yuen et S.C. Nyburg, "The matching of rigid bodies; a program using Kabsch's procedure", J. Appl. Cryst., **12**, 258 (1979)

4. E.T. Whittaker, "A Treatise on the analytical dynamics of particle and rigid bodies", Cambridge University Press, 2 (1970)

5. L. Barino, "Use of a least-squares best molecular fit routine in a steric comparison of flexible molecules", Comp. and Chem., **5**, 85 (1981)

6. (a) "Harwell subroutine library", Atomic energy research establishment, Harwell, Oxfordshire, England (1980)
   (b) M.J.D. Powell, "A fast algorithme for nonlinearly constrained optimization calculations", at the 1977 Dundee conference on numerical analysis. The proceedings will be published by Springer-Verlag in their series "Lecture notes in mathematics".

7. "Harwell subroutine library", Atomic energy research establishment, Harwell, Oxfordshire, England (1980) 7.

8. Jorge J. More, "The Levenberg-Marquard algorithm, implementation and theory. Numerical analysis", G. A. Watson, editor. Lecture notes in Mathematics 630, Springer-Verlag, 1977.

9. L. Solomon, M. Hocquemiller, "Mathematiques appliquees et calculatrices programmables: notation algébrique", Masson (1982)

10. D. Wexler, "Analyse : notes de cours", F.N.D.P. Namur (1981)

11. F. Callier,"Théorie des graphes : notes de cours", F.N.D.P. Namur

12. J. Lejeune, A. Michel, D.P. Vercauteren, "Improving the flexible molecular fitting technique using distance matrices", J. Comp. Chem., **7**, 739 (1986)

13. J. Lejeune, A. Michel, D.P. Vercauteren, "Interactive flexible molecular fitting program to be integrated into computer-aided molecular modelling systems", J. Mol. Graph., **4**, 194 (1986)

14. M. Bershon, "A fast algorithm for calculation of the distance matrix of a molecule", J. Comp. Chem., **5**, 110 (1983)

15. R. Tarjan, "Depth-first search and linear graph algorithms", SIAM J. Comput. , **1**, 146 (1972)

16. S.W. Golomb et L.D. Baumert, "Backtrack programming", J. ACM, **12**, 516 (1965)

17. N.J. Nilson, "Problem solving methods in artificial intelligence", McGraw-Hill, New-York (1971)